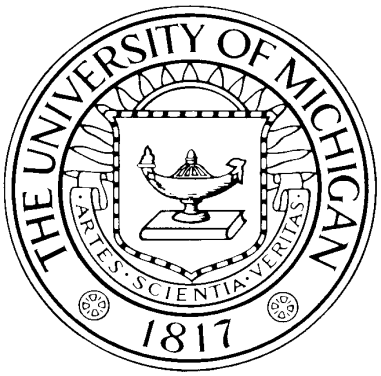
An Overview of the EPIC Architecture for Cognition and Performance with Application to Human-Computer Interaction

David E. Kieras
Artificial Intelligence Laboratory
Electrical Engineering & Computer Science Department
University of Michigan
1101 Beal Avenue, Ann Arbor, MI 48109-2110

and

David E. Meyer
Department of Psychology
University of Michigan
525 East University, Ann Arbor, MI 48109-1109



University of Michigan

EPIC Report No. 5 (TR-95/ONR-EPIC-5)

December 5, 1995

This research was supported by the Office of Naval Research, Cognitive Science Program, under Grant Number N00014-92-J-1173, Grant Authority Identification Number NR 4422574. Reproduction in whole or part is permitted for any purpose of the United States Government.

Approved for Public Release; Distribution Unlimited

An Overview of the EPIC Architecture for Cognition and Performance with Application to Human-Computer Interaction

David E. Kieras and David E. Meyer

University of Michigan

ABSTRACT

EPIC (**E**xecutive **P**rocess-**I**nteractive **C**ontrol) is a cognitive architecture especially suited for modeling human multimodal and multiple-task performance. The EPIC architecture includes peripheral sensory-motor processors surrounding a production-rule cognitive processor, and is being used to construct precise computational models for a variety of HCI situations. Some of these models are briefly illustrated here to demonstrate how EPIC clarifies basic properties of human performance and provides usefully precise accounts of performance speed and accuracy.

INTRODUCTION

This paper provides an overview of the EPIC (**E**xecutive **P**rocess-**I**nteractive **C**ontrol) architecture being developed by Kieras and Meyer for modeling human cognition and performance (Meyer & Kieras, in press-a, b, 1995; Kieras, Wood, & Meyer, 1995a,b; Kieras & Meyer, 1995). EPIC is similar in spirit to the Model Human Processor (MHP) (Card, Moran, & Newell, 1983), but EPIC incorporates many recent theoretical and empirical results about human performance in the form of a software framework for computer simulation modeling. Using EPIC, a model can be constructed that represents the general procedures required to perform a complex multimodal task as a set of production rules. When the model is supplied with the external stimuli for a specific task, it will then execute the procedures in whatever way the task requires, thus simulating a human performing the task, and generating the predicted actions in simulated real time. EPIC is not yet a learning system, and so has no mechanisms for learning how to perform a task; rather the purpose of EPIC is to represent in detail the perceptual and motor, as well as cognitive, constraints on human ability to perform tasks.

Like most cognitive architectures, EPIC was not developed primarily for addressing HCI problems, but is a larger scientific endeavor to represent important theoretical concepts of human intelligence or abilities. However, since HCI is a subset of human performance, a good proposal for a cognitive architecture should allow one to analyze and compare interface designs and then recommend and evaluate improvements. At this time EPIC is mainly useful as a research system for exploring human performance limitations that determine the effects of a particular interface design, both at low levels of specific interaction techniques, and at high levels of systems that support complex task performance in multimodal time-stressed domains. In the future it should be possible to develop design analysis techniques based on EPIC that can be routinely applied in system design.

The organization of the present paper is as follows: The rationale for the development of EPIC will be described, followed by a summary of the architecture, its assumptions and a discussion of some important modeling issues that apply not just to EPIC but to other efforts to represent human cognition and performance with computational models. EPIC's contributions to HCI will be illustrated with a series of brief examples showing how EPIC can be used in both a predictive and explanatory mode, to address both elementary aspects of interface design, and complex phenomena related to human interaction with semi-automated systems.

FUNDAMENTAL MOTIVATIONS FOR DEVELOPING EPIC

Embodied Cognition

The major cognitive architectures proposed thus far in cognitive psychology and artificial intelligence have been disembodied in that the human is represented as a purely cognitive system, a disembodied intelligence that directly perceives and acts on its environment. Such an approach can be seriously misleading. For example, human visual capacity to detect and recognize objects is not uniform, but varies with the distance on the retina from the fovea. The retina can be oriented through a motor system, the oculomotor mechanism, to control what part of the visual environment can be accessed in detail. The details of when, how, or whether the eye is oriented during task performance have been rarely considered.

Likewise conventional theory has tended to assume that once the human decides to act, there are no fundamental problems in carrying out the intended action. However, the human motor system is quite complex in its own right, and interacts strongly with the cognitive and perceptual systems (Rosenbaum, 1991). For example, different movements can take a substantial amount of time to execute, and this time can depend heavily on the details of the required movements and the history of previous, possibly interfering, movements. Furthermore, response execution can make demands on the perceptual system as well, the most extreme being the need for vision in aimed movements; thus, making a response may interfere with collecting the visual information needed for the next part of the task.

For these reasons, a primary motivation in developing EPIC has been to incorporate key facts about human perceptual and motor constraints into a theory of human cognition. Thus, EPIC's production-rule cognitive processor is surrounded by perceptual and motor processors, whose time course of processing is represented in some detail based on the current human performance research literature. How long it takes an EPIC model to do a task depends intimately on how EPIC's eyes, perceptual mechanisms, and effectors are used in the task. At this level of detail, the interactions between processors during task execution can be remarkably subtle, so the representation of task timing in a computational simulation model is critical to understanding human performance.

Computational Models of Both Performance and Cognition

A second motivation for developing EPIC has been to extend the computational modeling paradigm to the field of human attention and performance. This field has suffered from a lack of computational modeling, although historically, it is one of most extensively researched and most practically useful of psychological fields. Most research on human performance has been conducted at the level of qualitative interpretation of empirical results, and verbally-expressed and evaluated theory. Key issues concerning the fundamentals of human information processing have remained unresolved for many years, such as the status of the single-channel vs. multiple-resource debate discussed in Meyer and Kieras (in press-a, b). Thus, another goal of developing EPIC has been to advance the state of psychological theory in a theoretically underdeveloped area.

Our models are built by "programming" a fixed and comprehensive architecture with a task strategy expressed in production rules executed by the cognitive processor. EPIC shares this basic feature with other production-system architectures, but it is perhaps more distinctive in EPIC because the architecture itself is more comprehensive, including more than just the cognitive system. In the development of EPIC, this comprehensive architecture is typically treated as being fixed across a variety of task domains. We always attempt to explain phenomena in terms of task-specific cognitive strategies before changing the architecture itself.

An important benefit of working in the human performance domain is that we can use detailed quantitative fits to data as both additional constraints and as a goal for the modeling work. Unlike many modeling efforts, ours is

committed to obtaining detailed and quantitatively accurate fits to empirical data, and thus far have achieved good success because in the human performance domain, data typically have a precision and detail that is not found in more purely cognitive task domains. One reason for making this effort is that for models of performance to be practically useful in system design problems, they must be reasonably accurate. A more fundamental reason is that detailed and quantitatively accurate fits serve as a powerful constraint in constructing models for phenomena. Given that EPIC's architecture provides a fixed set of mechanisms with mostly fixed timing properties, only certain task strategies will work to match specific data. Thus the number of arbitrary decisions to be made in constructing an EPIC model for a task is sharply reduced. The constraints imposed by the combination of the detailed quantitative effects in the empirical data, the task structure, and the fixed architecture mean that there are relatively few "degrees of freedom" in model construction.

The Executive Process and Multiple-Task Performance

A third motivation for developing EPIC has been to explore the mechanism and the role of executive processes, which control and supervise other cognitive processes, analogous to the "supervisor" in a computer operating system. Theorists have presented various proposals about the nature of the executive process, but all have lacked rigor or computational representation, and have often proposed that the executive process is implemented via some type of special mechanism. However, since proper supervision of behavior is simply a form of skill, we have sought to represent the executive process in the same way as other forms of skill, just as the supervisory component of a computer operating system is just another computer program.

The best approach to understanding both the nature of the executive process and the details of the human cognitive architecture is to understand multiple-task performance. In multiple-task situations, the human has to perform two or more tasks simultaneously; the overall task situation can be subdivided into two or more tasks, each of which can be meaningfully performed in isolation (one is not a logical subtask of the other) and the tasks are performed over the same period of time. A good example of a multiple-task situation is an airplane cockpit; a pilot may need to simultaneously pilot the aircraft and track an enemy target on a radar display. The main problem confronting the human is to execute the independent tasks in a coordinated fashion that meets some constraints on overall performance, such as giving one task priority over the other.

The literature on multiple-task performance is extensive, and will not be summarized here; for a review, see Gopher and Donchin (1986). Of course, human information processing is limited in capacity, and it has been traditionally assumed that there is a single-channel bottleneck (Welford, 1952). Nevertheless, humans can do multiple tasks, sometimes impressively well, and their ability to do so depends strongly on the specific combinations of tasks involved. The multiple-resource theory (Wickens, 1984) is an attempt to summarize these dependencies. They pose a fundamental theoretical dilemma about how to reconcile the complex patterns of people's multitasking abilities with some notion that the overall capacity of the human system is limited. However, EPIC does not assume that central capacity for cognitive processing is limited. Such an assumption is traditional, but lacks both empirical and metatheoretical justification. In contrast, we have assumed that limitations on human ability are all structural; that is, performance of tasks may be limited by constraints on peripheral perceptual and motor mechanisms, rather than a pervasive limit on cognitive processing. Thus, the executive strategy has the responsibility of meeting the performance requirements of the tasks in spite of these structural limitations.

To meet performance goals, the executive process must coordinate the use of the perceptual, cognitive, and motor resources of the system so that the tasks can be conducted with the proper relative priority and speed. Multiple-task situations stress human capabilities very seriously, and so the observed patterns of behavior provide clear insights into the abilities and limitations of the human information-processing system architecture. Yet despite the practical importance of multiple-task performance, the empirical and theoretical understanding of them has been quite limited. Nevertheless, EPIC models have been successful at accounting with unprecedented accuracy for performance in laboratory versions of multiple-task situations (Meyer & Kieras, 1995, in press-a, b; Kieras & Meyer, 1995). In

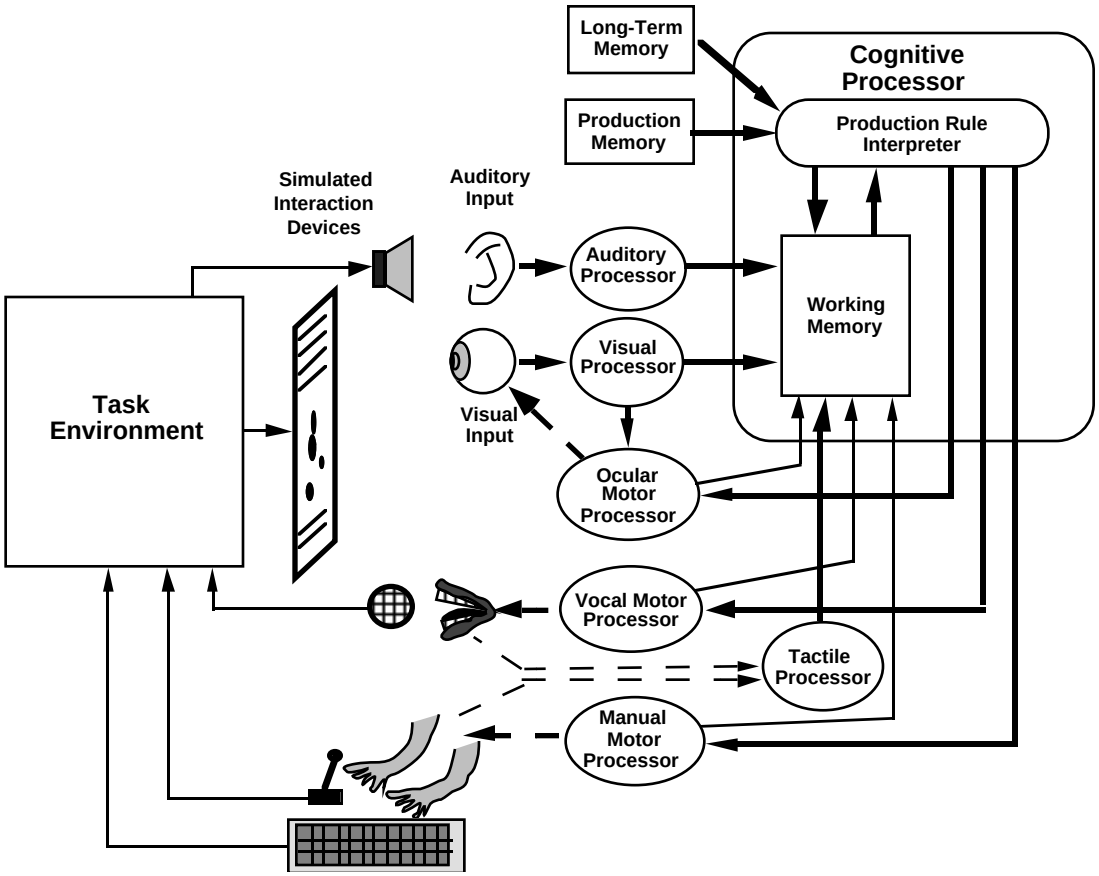


Figure 1. Overall structure of the EPIC architecture simulation system. Task performance is simulated by having the EPIC model for a simulated human (on the right) interact with a simulated task environment (on the left) via a simulated interface between sensory and motor organs and interaction devices. Information flow paths are shown as solid lines, and mechanical control or connections as dashed lines. The processors run independently and in parallel with both each other and the Task Environment module.

addition, understanding the allocation of resources in multiple task situations contributes to understanding single-task situations; to maximize performance, the executive process must allocate processing resources to different parts of the single task in a properly coordinated fashion.

THE EPIC ARCHITECTURE

Figure 1 shows the overall structure of processors and memories in the EPIC architecture. Although at this level EPIC bears a superficial resemblance to earlier frameworks for human information processing, EPIC incorporates a new synthesis of theoretical concepts and empirical results, and so is more comprehensive and more detailed than earlier proposals for human performance modeling such as MHP, HOS, SAINT, and so forth (see McMillan, Beevis, Salas, Strub, Sutton, & Van Breda, 1989). EPIC is designed to explicitly couple detailed mechanisms for basic information processing and perceptual-motor activity with a cognitive analysis of procedural skill, namely that represented by production-system models such as CCT (Bovair, Kieras, & Polson, 1990), ACT-R (Anderson, 1993), and SOAR (Laird, Rosenbloom, & Newell, 1986). Thus, EPIC has a production-rule cognitive processor

surrounded by perceptual-motor peripherals; applying EPIC to a task situation requires specifying *both* the production-rule programming for the cognitive processor, and also the relevant perceptual and motor processing parameters. EPIC computational task models are *generative* in that the production rules supply procedural knowledge of the task, and when EPIC interacts with a simulated task environment, the EPIC model generates the specific sequence of serial and parallel human actions required to perform the specific tasks. The task analysis reflected in the model is general to a class of tasks, rather than reflecting specific task scenarios.

The software for constructing EPIC models includes not only the modules for simulating a human, but also facilities for simulating the interaction of the human with an external system such as a computer. Figure 1 shows at the left a simulated task environment, and on the right, a simulated human as described by the EPIC architecture, with objects such as screen items and keys making up the physical interface between them. The task environment module assigns physical locations to the interface objects, and generates visual events and sounds that the computer or other entities in the environment produce in response to the simulated human's behavior. Having a separate environment simulation module greatly simplifies the programming of a complete simulation, and helps enforce the generality of the procedural knowledge represented in the EPIC model. That is, the task environment module is driven by a task instance description that consists only of the sequence and timing of events external to the human user, and the simulated user must deal with whatever happens in the simulated task environment.

With regard to the EPIC architecture itself as shown in Figure 1, there is a conventional flow of information from sense organs, through perceptual processors, to a cognitive processor (consisting of a production rule interpreter and a working memory), and finally to motor processors that control effector organs. EPIC goes beyond the MHP by specifying separate perceptual processors with distinct processing time characteristics for each sensory modality, and separate motor processors for vocal, manual, and oculomotor (eye) movements. There are feedback pathways from the motor processors, as well as tactile feedback from the effectors, which are important in coordinating multiple tasks. The declarative/procedural knowledge distinction of the "ACT-class" cognitive architectures (e.g., Anderson, 1976) is represented in the form of separate permanent memories for production rules and declarative information. Working memory (WM) contains all of the temporary information needed for and manipulated by the production rules, including control items such as task goals and sequencing indices, and also conventional working memory items, such as representations of sensory inputs. Each of these processors will be described in more detail subsequently. When numeric values are given for various time parameters, they will be labeled as either standard values that we assume are fixed in all applications of the architecture, or as typical values, which may vary depending on the properties of a specific task situation. The standard values are based on our reading of the human performance literature; while they may be revised and refined, they are supposed to hold across all applications of the architecture. The non-standard parameter values need to be estimated to model a specific task, but we hope that with additional modeling experience and focused empirical studies, collections of typical parameter values will become available for use in constructing new models.

Perceptual Processors

A single stimulus input to a perceptual processor can produce multiple outputs to be deposited in WM at different times. The perceptual processors in EPIC are simple "pipelines," in that an input produces an output at a certain later time, with no "moving window" time-integration effect as assumed by the MHP. The tactile perceptual processor handles movement feedback from effector organs; this feedback can be important in coordinating multiple tasks (Meyer & Kieras, in press-a, b), but is not elaborated further here.

Visual processor. EPIC's model of the eye includes a retina that determines what kind of sensory information is available about visual objects in the environment based on the distance (in visual angle) on the retina between the object and center of the fovea. EPIC's current highly simplified model of the retina contains three zones, each with a standard radius: the fovea (1°), the parafovea (10°), and the periphery (60°). Certain information, such as the contents of character strings, is available only in the fovea, whereas cruder information, such as whether an area of

the screen is filled with characters, is available in the parafovea. Only severely limited information is available in peripheral vision, such as the location of objects, and whether an object has just appeared. In EPIC's visual working memory, the visual perceptual processor maintains a representation of which objects are visible and what their properties are. Visual working memory is "slaved" to the visual situation; it is kept up-to-date as objects appear, disappear, change color, and so forth, or as eye movements or object movements change what visual properties are available from the retina. In response to visual events, the visual processor can produce multiple outputs with different timings. When an object appears, the first output is a representation that a perceptual event has been detected (standard delay: 50 ms), followed later by a representation of sensory properties (e.g., shape, standard delay: 100 ms), and still later by the results of pattern recognition, which might be task-specific (e.g., a particular shape represents a left-pointing arrow, typical delay: 250 ms). The exact time delays and radii of availability depend on a variety of sensory properties and their relationships; because the relevant data on human performance are quite patchy, many of parameters of visual processing must be estimated for new task-specific models.

Auditory processor. The auditory perceptual processor accepts auditory input, and outputs to working memory representations of auditory events (e.g., speech) that disappear after a time. For example, a short tone signal first produces an item corresponding to the onset of the tone (standard delay: 50 ms), then at a later time, an item corresponding to a discriminated frequency of the tone (typical delay: 250 ms), then an offset item (standard delay: 50 ms). For simplicity such items simply disappear from memory after a fixed time (typical delay: 4 s).

Speech input is represented as items for single words in auditory working memory. The auditory perceptual processor requires a certain time to recognize input words (typical delay: 150 ms after acoustic stimuli are present) and produces representations of them in auditory working memory. These items then disappear after a time, the same as other auditory input. To represent the sequential order of the speech input, the items contain arbitrary symbolic tags for the previous and the next item that link the items in sequence. Thus, a speech input word carries a certain next-tag value, and the next word in the sequence is the item that contains the same tag value as its previous-tag. Using these tags, a set of production rules can step through the auditory working memory items for a series of spoken words, processing them one at a time. For example, the models described in this paper process a spoken telephone billing number by retrieving the recognized code for each digit in the tag-chained sequence and using it to specify a key press action. Available empirical literature on auditory perception lacks comprehensive results, so many auditory perceptual parameters must be estimated during task-specific model construction.

Cognitive Processor

Production rules and cycle time. The cognitive processor is programmed in terms of production rules, and so an EPIC model for a task must include a set of production rules that specify what actions in what situations are performed to do the task. Example production rules for the models described in this paper will be presented later. EPIC uses the Parsimonious Production System (PPS) interpreter, which is especially suited to task modeling work (Bovair, Kieras, & Polson, 1990). PPS rules have the format (<rule-name> IF <condition> THEN <actions>). The rule condition can test only the contents of the production system working memory. The rule actions can add and remove items from the working memory, or send a command to a motor processor.

The cognitive processor operates cyclically. At the beginning of each cycle, the contents of working memory are updated with the output from perceptual processors, and the previous cycle's modifications; at the end of each cycle, the contents of the production system working memory are updated and commands are sent to the motor processors. The mean duration of a cycle is a standard 50 ms. The cognitive processor cycles are not synchronized with external stimulus and response events. Inputs from the perceptual processors are accessed only intermittently, when the production-system working memory is updated at the start of each cycle. Any input that arrives during the course of a cycle must therefore wait temporarily for service until the next cycle begins. This is consistent with a variety of phenomena, such as the apparent temporal granularity of perceived stimulus successiveness (Kristofferson, 1967).

EPIC also can run in a mode in which the cycle duration is stochastic, with a standard mean value of 50 ms and all other time parameters scaled to this stochastic value; the variance of the stochastic distribution of cycle time is chosen to produce a coefficient of variation of about 20% for a simple reaction time, corresponding to the typical observed value.

Unlike most other production-system architectures, on each cognitive processor cycle, PPS will fire all rules whose conditions match the contents of working memory and will execute all of their actions. Thus, EPIC models allow for true parallel cognitive processing; the EPIC cognitive processor is not constrained to be doing only one thing at time. Rather, multiple processing threads can be represented simply as sets of rules that happen to run simultaneously. The multiprocessing ability of the cognitive processor, together with the parallel operation of all the perceptual-motor processors, means that EPIC models for multiple-task performance do not incorporate a gratuitous assumption of limited central processing capacity. Rather, EPIC emphasizes the role of executive-process strategies in coordinating perceptual-motor peripherals to perform multiple tasks. As shown by detailed quantitative modeling of a variety of multiple-task data reported in Meyer and Kieras (in press-a, b) and Kieras and Meyer (1995), EPIC and its approach to modeling multiple-task performance has excellent empirical support; moreover, as argued by Meyer et al. (in press) and Meyer and Kieras (in press-a, b), the traditional assumption of limited central capacity lacks strong empirical support.

Working memory. EPIC's production-system working memory is in effect partitioned into several working memories.[1] Visual, auditory, and tactile working memory contain the current information produced by the corresponding perceptual processors. The timing and duration of these forms of working memory were described earlier. Motor working memory contains information about the current state of the motor processors, such as whether a hand movement is in progress. This information is updated on every cycle.

Two other forms of working memory deserve special note; these are *amodal* in that they contain information not directly derived from sensory or motor mechanisms. One amodal working memory is the control store, which contains items that represent the current goals and the current steps within the procedures for accomplishing the goals. An important feature of PPS is that control information is simply another type of working memory item, and so can be manipulated by rule actions; this is critical for modeling multiple-task performance, in that production rules for an executive process can control subprocesses by manipulating the control store (see Meyer & Kieras, in press-a, b, for more discussion).

The second amodal working memory, simply termed "general WM," can be used to store miscellaneous task information. At this time EPIC does not include assumptions about the decay, capacity, and representational properties of general working memory. Our research strategy in developing EPIC has been to see what constraints on the nature of WM are required to model task performance in detail, rather than following the customary strategy in cognitive modeling of assuming these constraints in advance. Such capacity and loss assumptions for these memory systems do not seem to be required to account for the time course of performance in tasks modeled in EPIC thus far; other limitations determined by the perceptual and motor systems appear to dominate performance. These latter substantial but under-appreciated limitations would have been obscured by gratuitous assumptions about central processing capacity or working memory (see Meyer & Kieras, in press-a, b, for more discussion). For similar reasons, at this time EPIC assumes that information is not lost from the control store, and there is no limit on the capacity of the control store. Research is underway to explore how loss or corruption of information in EPIC's working memories might account for the occurrence and properties of human errors during task performance.

[1] EPIC's working memory structure is not "hard-wired" into PPS. PPS actually has only a single "working me which could more clearly be termed the "database" for the production rules. PPS can be used as a multiple-memory simply by following the convention that the first term in database items indicates the "type" of memory item, as in examples later in this paper.

Motor Processors

The EPIC motor processors are much more elaborate than those in the MHP, producing a variety of simulated movements of different effector organs, and taking varying amounts of time to do so. As shown in Figure 1, there are separate processors for the hands, eyes, and vocal organs, and all can be active simultaneously. The cognitive processor sends a command to a motor processor that consists of a symbolic name for the type of desired movement and any relevant parameters, and the motor processor then produces a simulated movement with the proper time characteristics. The various processors have similar structures, but different timing properties and capabilities, based on the current human performance literature in motor control (Rosenbaum, 1991). The manual motor processor has many movement forms, or styles, but the two hands are bottlenecked through a single manual processor, and so normally can be operated either one at a time, or synchronized with each other. The oculomotor processor generates eye movements either upon cognitive command, or in response to certain visual events. The vocal motor processor produces a sequence of simulated speech sounds given a symbol for the desired utterance.

Movement preparation and execution. The variou motor processors represent movements and movement generation in the same basic way. Current research on movement control (Rosenbaum, 1980, 1991) suggests that movements are specified in terms of movement features, and the time to produce a movement depends on its feature structure as well as its mechanical properties.

The overall time to complete a movement can be divided into a preparation phase and an execution phase. The preparation phase begins when the motor processor receives the command from the cognitive processor. The motor processor recodes the name of the commanded movement into a set of movement features, whose values depend on the style and characteristics of the movement, and then generates the features, taking a standard 50 ms for each one. The time to generate the features depends on how many features can be reused from the previous movements (repeated movements can be initiated sooner), and how many features have been generated in advance. Once the features are prepared, the execution phase begins with an additional standard delay of 50 ms to initiate the movement, followed by the actual physical movement. The time to physically execute the movement depends on its mechanical properties, both in terms of which effector organ is involved (e.g., the eye versus the hand) and the type of movement to be made (e.g., a single finger flexion to press a button under the finger versus a pointing motion with a mouse).

The movement features remain in the motor processor's memory, so future movements that share the same features can be performed more rapidly. However, there are limits on whether features can be reused; for example, if a new movement is different in style from the previous movement, all of its features must be generated anew. Also, if the task permits the movement to be anticipated, the cognitive processor can command the motor processor to prepare the movement in advance by generating all of the required features and saving them in motor memory. Then when it is time to make the movement, only the initiation time is required to commence the mechanical execution of the movement.

Finally, a motor processor can prepare the features for only one movement at a time, and will reject any subsequent commands received during the preparation phase, but the preparation for a new movement can be done in parallel with the physical execution of a previously commanded movement. Once prepared, the movement features are saved in motor memory until the previous execution is complete, and the new movement is then initiated. The cognitive-processor production rules can exploit this capability by sending a motor processor a new movement command as soon as it is ready to begin preparing the features for the new movement. The result can be a series of very rapid movements whose total time is little more than the sum of their initiation and mechanical execution times.

An example of motor processor operation. To strike a key using a one-finger peck movement style (like that used in "hunt-and-peck" typing), the cognitive processor commands the manual motor processor to perform a peck movement with some finger (e.g., the right index) to a specified object in the physical environment (the key). This movement style involves five features: the peck style, the hand, the finger, the direction, and the extent of the motion, which is the distance between the current location of the designated finger and the location of the target object. If a previous movement was also a peck movement with the same hand and finger, only the direction and extent might have to be generated anew. If the movement is also similar in direction and extent to the previous movement, then all of the features could be reused; none would have to be generated anew. Once the features are generated, the movement is initiated. The time required to physically execute the movement to the target is given by Welford's form of Fitts' law (see Card, Moran, Newell, 1983, Ch. 2), with a standard minimum execution time of 100 ms, reflecting that for small movements to large targets, there is a physiologically determined lower bound on the actual duration of a muscular movement. After the simulated finger hits the key, it is left in the location above the key to await the next movement.

Manual motor processor. EPIC's manual motor processor represents several movement styles, including punching individual keys or buttons already known to be below the finger, pecking keys that may require some horizontal motion, posing the entire hand at a specified location, pattering two-fingers movements one after the other, pointing at an object, and plying a control (e.g., a mouse or joystick) to position a cursor onto an object. Each style of movement has a particular feature structure and an execution time function that specifies how long the mechanical movement takes to actuate the device in the task environment.

Vocal motor processor. EPIC's vocal motor processor is not very elaborated at this time; it is based on the minimal facilities needed to model certain dual-task situations (see Meyer & Kieras, in press-a, b). A more complete version of the vocal motor processor would be able to produce extended utterances of variable content, taking into account that the sequential nature of speech means that movements can be prepared on the fly during the ongoing speech. The current version of EPIC assumes that simple fixed utterances can be designated with a single symbol, and require only the preparation of two features before execution begins. The actual production of the sound is assumed to be delayed by about 100 ms after initiation, and continues for a time determined by the number of syllables in the words. Further development of the vocal motor processor is planned for the future.

Oculomotor processor. EPIC's eye movements are produced in two modes, voluntary and involuntary (reflexive). The cognitive processor commands voluntary eye movements, which are saccades to a designated object. A saccade requires generation of up to two features, a direction and extent of the movement from the current eye position to the target object. Execution of the saccade currently is estimated to require a standard 4 ms/degree of visual angle, which includes a residual component duration for settling on the movement endpoint. The oculomotor processor also makes involuntary eye movements, either saccades or small smooth adjustments in response to the visual situation (hence the arrow between the visual perceptual processor and the oculomotor processor in Figure 1). A sudden onset (appearance) of an object can trigger an involuntary saccade. The fovea being somewhat off-center on an object will produce a "centering" movement, which will automatically help zero-in on an object after a saccade. Also, the eye will automatically follow a slowly moving object using smooth movements and occasional small saccades (cf. Hallett, 1986). In the tasks considered in Kieras and Meyer (1995), EPIC can follow moving objects with a mixture of voluntary and involuntary eye movements. The partial autonomy of the oculomotor processor permits the cognitive processor to choose an object to examine, and then leave the details of keeping it centered on the fovea to the oculomotor processor.

Constraints in Model Construction

Fixed and free parameters. The presentation of any modeling approach should document what aspects or parameters of the modeling framework are fixed and are thus supposed to generalize across applications, and what parameters have to be estimated from data specific to the situation being modeled. In EPIC, the fixed aspects and

parameters are: (1) the connections and mechanisms of the EPIC processors; (2) most time parameters in the processors; and (3) the feature structure of the motor processors. Thus, adopting the EPIC framework entails committing to all these details of the modeling framework. The model properties and parameters that are then free to vary from model to model or task to task are: (1) the task-specific production-rule programming, which is constrained to some extent because it must be written to execute the task correctly and reasonably efficiently; (2) the task-specific sensory availabilities and perceptual encoding types and times involved in the task; these must be defined in terms of the production rules, and are constrained to be similar and constant over similar perceptual events; (3) the styles of movement used to control the device (e.g., touch-typing versus visually-guided pecking), if it is not constrained by the task.

Model inputs and outputs. Similarly, any modeling approach should document what information the model builder has to supply in order to construct the model, and what information the constructed model will then produce in return for the supplied information. To construct an EPIC model, the model builder has to supply the information corresponding to the free parameters described above, namely: (1) a production-rule representation of the task procedures; (2) task-specific sensory availabilities and perceptual-processor encodings and timings; and (3) any movement styles not determined by the task requirements. In addition, the model builder must supply: (4) the simulated task environment, which includes the physical locations and characteristics of relevant objects external to the human; and (5) a set of task instances whose execution time is of interest; these instances must specify only environmental events and their timing, and are used to control only the environment module of the simulation.

In return for these inputs, the EPIC model will interact with the simulated task environment, generating the predicted sequences of simulated human actions required by each task instance, and the predicted time of occurrence of each action. If the production rules were written to describe general procedural knowledge of how to perform the task, these predictions can be generated for any task instance subsumed by these general procedures.

Multiple Tasks and Executive Processes

In developing EPIC, our theoretical strategy has been to make some radical simplifying assumptions and then explore their consequences through modeling. We assume that all capacity limitations are a result of limited structural resources, rather than a limited cognitive processor. The EPIC cognitive processor can fire any number of rules simultaneously, but since the peripheral sense organs and effectors are structurally limited, the overall system is sharply limited in capacity. For example, the eyes can only fixate on one place at a time, and the two hands are bottlenecked through a single processor. Thus, a task that demands manual responses to visual stimuli distributed over a wide space will result in severely limited human performance, even if the purely cognitive demands are trivial. We also assume that certain apparent limitations in central capacity arise when modality-specific working memories must be used to maintain task information, but we have not yet tested this assumption in the EPIC framework. Thus far, our simple and radical set of assumptions about the nature of multiple-task processing limitations has held up well.

Some theories of multiple-task performance postulate an executive control process that coordinates the separate multiple tasks (e.g. Norman & Shallice, 1986). We do likewise, but a key feature of our approach is that the executive control process is just another set of production rules. These rules can control other task processes by manipulating information in WM. For example, we assume that each task is represented by a set of production rules that have the task goal appearing in their conditions, and so an executive-process rule can suspend a task by removing its governing goal from WM, and then cause it to resume operation by reinserting the goal in WM. Also, the executive process can cause a task to follow a different strategy by placing in WM an item that task rules test for, thus enabling one set of rules, and disabling another. In addition, the executive process may control sensory and motor peripherals directly, such as moving the eye fixation from one point to another, in order to allocate these resources between two tasks. Thus, rather than postulating an executive control mechanism that is somehow different in kind than other cognitive mechanisms, EPIC has a uniform mechanism for the control of

behavior, both at the executive level and at the detailed level of individual task actions. As a corollary, learning how to coordinate multiple tasks is simply learning another (possibly difficult) skill, as has been proposed by some recent investigators (Gopher, 1993).

Previous Work on Multiple-Task Performance

Our first work with EPIC has focused on the simplest and most heavily studied dual-task situation in the research literature, the so-called Psychological Refractory Period (PRP) procedure. The PRP procedure consists of two temporally overlapping choice reaction-time tasks; the subject is instructed to make the response for the first task before making the response for the second task. The primary measure of interest is the reaction time for the second task, which may be affected by the temporal spacing between the two task stimuli. The basic empirical result is that the second response is substantially delayed as the spacing between the two stimuli decreases. The conventional interpretation of this effect (the PRP effect) is that the human has a central response-selection bottleneck, and so the second response cannot be selected or initiated until the first response has been made. However, the details of the effect, and how it depends on other factors such as the stimulus and response modalities of the two tasks, form a complex pattern that has never been satisfactorily explained in any detail.

Meyer and Kieras (in press-a, b) provide an exhaustive treatment of the PRP effect using EPIC simulations, and mathematical analyses based on them, to account quantitatively for the results in many published experiments, and in new experiments (Meyer et al., in press). The correct interpretation of the PRP effect is that in order to conform to the task instructions, subjects must adopt a strategy that postpones initiating the second response until they can ensure that it does not occur before the first response; the magnitude of the delay in the second response depends on how much of the second-task processing can be overlapped with the first task, which in turn depends on the details of the task structure (e.g. whether eye movements are required), the task difficulty, and the task modalities. The EPIC architecture captures the relevant constraints very well; Meyer and Kieras were able to construct models that accounted for the specific patterns of effects in quantitative detail, and revealed the underlying structure of the phenomena. More details are available in Meyer and Kieras (in press-a, b; Meyer et al., in press).

Lessons from multiple-task modeling. An immediate insight from the application of the EPIC architecture to multiple-task domains is that there are many possibilities for performing task activities in parallel. That is, in dual-task models using EPIC, the role of the cognitive strategies in coordinating activity between the two tasks is critical to accounting for the observed effects, and in many situations, these strategies are surprisingly subtle and efficient. In dual-task experiments, the subject is supposed to complete each of two tasks as rapidly as possible, but the higher-priority task must be completed before the lower-priority task, regardless of the relative speed of the perceptual or motor processing involved in the two tasks. If two tasks require the same motor processor, both perceptual and cognitive processing on the lower-priority task can go on while the higher-priority task is allowed to control the motor processor. Once the motor processor has commenced execution of a higher-priority response, the lower-priority task can be given control of the motor processor, thereby honoring the task coordination requirements while maximizing speed. If the two tasks involve different motor modalities, portions of the lower-priority response can be prepared far in advance, so that it can be made more quickly when its turn comes. If the two tasks compete for the use of both the eyes and the hands, the executive rules can dynamically switch control of the two processors between the two tasks so that their processing is interleaved, with little wasted time. Thus, in a dual-task situation, the cognitive processor strategies are responsible for allocating the eyes and the motor processors to the two tasks as needed to maximize overall performance. Similarly, in a single multimodal task with a requirement for speed, the task strategy is responsible for ensuring that the individual processors do their work as soon as possible so as to minimize the total time required for the task.

The need for modeling policies. There are multiple possibilities for how activities in a multimodal task can be overlapped under the EPIC architecture. One way to identify the specific strategy that governs overlapping in a task is to propose a strategy, generate predicted performance under that strategy, compare the predicted performance

to empirical data, and repeat until the predicted data match the empirical results. In typical scientific cognitive modeling work devoted to verifying a cognitive architecture and understanding how a task might be done, it is acceptable to arrive at task strategies in this post-hoc model-fitting mode. However, once scientific work on cognitive architectures has progressed beyond simple demonstrations of feasibility, success at *a-priori* prediction is required to fully establish the architecture on scientific grounds and to use the architecture in practical system design to analyze the merits of alternative designs. Predicting performance on an *a-priori* basis requires not only a usefully accurate cognitive architecture, but also a set of modeling policies for how to choose and represent task strategies on an *a-priori* basis. Developing such policies can only be done by systematically characterizing the space of possible models and testing their accuracy; an example appears in Kieras, Wood, and Meyer (1995a, b).

In Kieras, Wood, and Meyer (1995a, b), EPIC was used to predict performance in a well-practiced, multimodal task, that of telephone operators, who collect billing numbers spoken by customers, enter them into a computer workstation, and verify the number before allowing the call to proceed. The volume of this work is such that saving a few seconds of work time per call is worth millions of dollars annually in labor costs. Human operators

```
(IF-NOT-TARGET-THEN-SACCADE-ONE-ITEM
IF
((GOAL DO MENU TASK)
 (STEP VISUAL-SEARCH)
 (WM CURRENT-ITEM IS ?OBJECT)
 (VISUAL ?OBJECT IS-ABOVE ?NEXT-OBJECT)
 (NOT (VISUAL ?OBJECT IS-ABOVE NOTHING))
 (MOTOR OCULAR PROCESSOR FREE)
 (VISUAL ?OBJECT LABEL ?NT)
 (NOT (WM TARGET-TEXT IS ?NT)))
THEN
((DELDB (WM CURRENT-ITEM IS ?OBJECT))
 (ADDDB (WM CURRENT-ITEM IS ?NEXT-OBJECT))
 (SEND-TO-MOTOR OCULAR MOVE ?NEXT-OBJECT)))

(TARGET-IS-LOCATED-BEGIN-MOVING-MOUSE
IF
((GOAL DO MENU TASK)
 (STEP VISUAL-SEARCH)
 (WM TARGET-TEXT IS ?T)
 (VISUAL ?TARGET-OBJECT LABEL ?T)
 (WM CURSOR IS ?CURSOR-OBJECT)
 (MOTOR MANUAL PROCESSOR FREE))
THEN
((DELDB (STEP VISUAL-SEARCH))
 (ADDDB (STEP MAKE RESPONSE))
 (SEND-TO-MOTOR MANUAL PERFORM PLY MOUSE
       RIGHT ZERO-ORDER-CONTROL ?CURSOR-OBJECT ?TARGET-OBJECT)))
```

Figure 2. Productions rules from the serial search model of menu selection. The first rule repeatedly moves the eye down the menu as long as the text label for the menu item fails to match the sought-for item in working memory. If a matching item is found, the second rule points the mouse cursor to it.

normally overlap speaking and listening to the customer with pressing keys and watching for information to appear on the screen. The time taken to handle the call is not simply the sum of the individual activity times, but is a complex function of which activities can be overlapped and to what extent. Our EPIC models were constructed on

an *a-priori* basis following several modeling policies that start with a simple procedural task analysis which is then translated in a standardized format into a set of EPIC production rules. The predicted task execution times were accurate within 10-14%, which is useful in an engineering context. The EPIC architecture accurately represents the perceptual and motor constraints in the task, making it possible to easily construct a model on an *a-priori* basis that predicts the task time accurately enough to aid in choosing between alternative designs. The effort required to construct the EPIC models is fairly modest. In return, the resulting EPIC models can generate predicted execution times for all possible task instances within the scope of the model. Thus, EPIC models appear to be efficient engineering models for multimodal high-performance tasks.

EXAMPLES OF APPLYING EPIC TO HCI

This section presents illustrative applications of EPIC to various HCI situations, ranging from low-level interaction phenomena, to complex interactions with visual displays in dual-task settings. Two recurring themes in this work are the critical role of visual layout and eye movements, and the importance of parallel processing or multitask execution, even in simple situations. Each example application is described only briefly; more complete work is in progress.

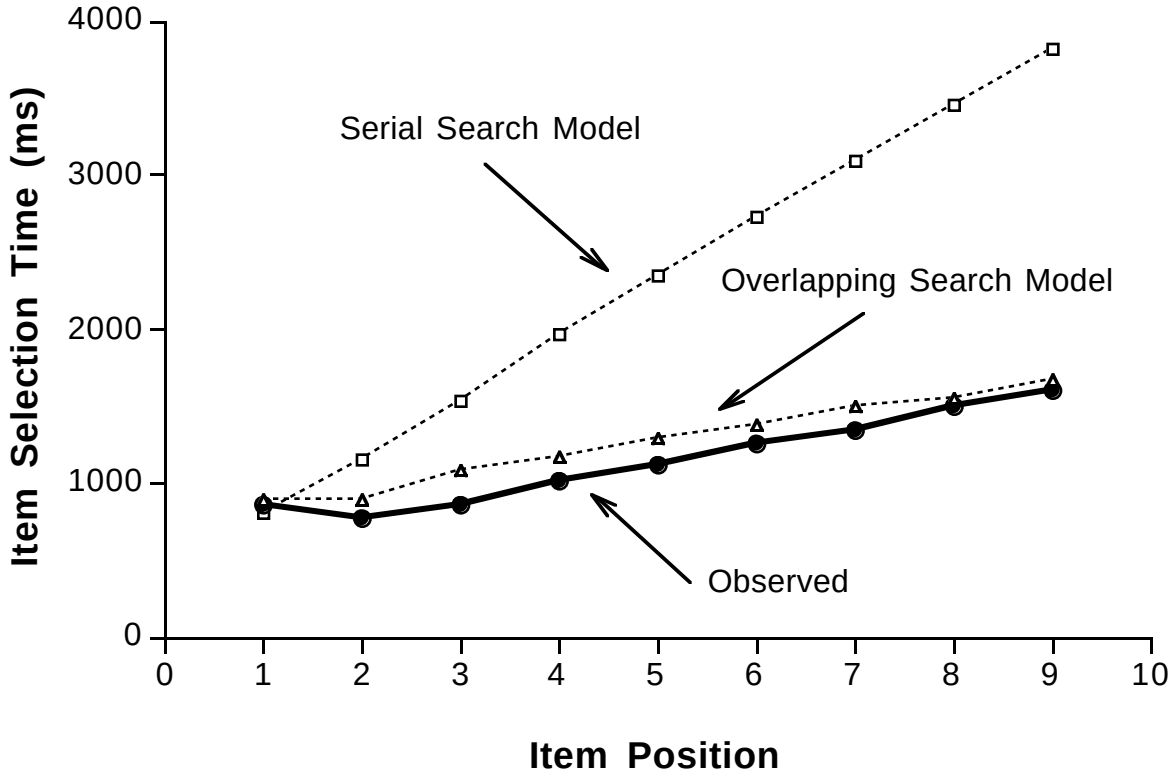


Figure 3. Observed and predicted menu selection times. Observed data is from Nilsen (1991); predicted results are explained in the text.

Selecting Items from Menus.

Choosing items from a pull-down menu with a mouse is a standard feature of many current interfaces. However, the research and practical design literature does not contain a comprehensive or even very explicit model of how such menu access is done. At first glance, it would seem that the user must first visually scan the list of menu items, looking at each one in turn, and when the desired item is found, make a movement with the mouse to position the cursor there. Other authors (e.g. Sears & Shneiderman, 1994) have made other proposals for menu search that are rather more elaborate, but without detailed testing against empirical data. EPIC can be used to represent different

```
(SACCADE-ONE-ITEM
IF
((GOAL DO MENU TASK)
 (STEP VISUAL-SWEEP)
 (WM CURRENT-ITEM IS ?OBJECT)
 (VISUAL ?OBJECT IS-ABOVE ?NEXT-OBJECT)
 (NOT (VISUAL ?OBJECT IS-ABOVE NOTHING))
 (MOTOR OCULAR PROCESSOR FREE)
 )
THEN
((DELDB (WM CURRENT-ITEM IS ?OBJECT))
 (ADDDB (WM CURRENT-ITEM IS ?NEXT-OBJECT))
 (SEND-TO-MOTOR OCULAR MOVE ?NEXT-OBJECT)))

(STOP-SCANNING
IF
((GOAL DO MENU TASK)
 (STEP VISUAL-SWEEP)
 (WM TARGET-TEXT IS ?T)
 (VISUAL ?TARGET-OBJECT LABEL ?T))
THEN
((DELDB (STEP VISUAL-SWEEP))
 (ADDDB (STEP MOVE-GAZE-AND-CURSOR-TO-TARGET))
 (ADDDB (WM TARGET-OBJECT IS ?TARGET-OBJECT))))

(MOVE-GAZE-AND-CURSOR-TO-TARGET
IF
((GOAL DO MENU TASK)
 (STEP MOVE-GAZE-AND-CURSOR-TO-TARGET)
 (WM TARGET-OBJECT IS ?TARGET-OBJECT)
 (WM CURSOR IS ?CURSOR-OBJECT)
 (MOTOR OCULAR PROCESSOR FREE)
 (MOTOR MANUAL MODALITY FREE))
THEN
((DELDB (STEP MOVE-GAZE-AND-CURSOR-TO-TARGET))
 (ADDDB (STEP MAKE RESPONSE))
 (SEND-TO-MOTOR OCULAR MOVE ?TARGET-OBJECT)
 (SEND-TO-MOTOR MANUAL PERFORM PLY MOUSE
       RIGHT ZERO-ORDER-CONTROL ?CURSOR-OBJECT ?TARGET-OBJECT)))
```

Figure 4. Production rules from the Overlapping Search model. The first rule moves the eye rapidly down the menu, regardless of whether a matching item is present or not. The second rule halts the scan if a matching item appears in visual working memory, and enables the third rule, which moves the eye back to the matching item and launches the mouse pointing movement to it.

hypotheses about how users select menu items, which can then be compared to data with high precision. Working in our lab, Anthony Hornof has begun to model menu access as part of a larger program of research on visual search and visual layout. His first results suggest that EPIC can be used to address a variety of visual issues in interface design.

Hornof has modeled performance in a task that was one condition of an experiment by Nilsen (1991) in which subjects selected items from a pull-down menu. In Nilsen's experiment, the subject was first shown a digit, then clicked on a target. This would cause a vertical menu of the digits 1-9 to appear in a random order below the cursor position. The subject then pointed to and clicked on the previously designated digit in the menu. The time to select a digit as a function of its location in the randomly ordered menu was fairly linear with a slope of about 100 ms per item; similar results have been obtained elsewhere.

Serial search model. Hornof has constructed a *serial search* model for menu item selection that corresponds to the one-at-time hypothesis of visual search; the eye is moved to the next object down the menu, and if it matches the sought-for item, a pointing movement is initiated to the item. Otherwise, the eye is moved to the next object. Figure 2 shows the key production rules in this model. The rule `IF-NOT-TARGET-THEN-SACCADE-ONE-ITEM` waits for the current visual object (bound to the variable `?OBJECT`) to have a text `LABEL` property in visual working memory, and fires if this label does not match the sought-for label bound to the variable `?NT`. If the rule fires, the `DELDB` action (delete from the production system database) and `ADDDB` action (add to the database) actions update the production system database to make the object below the current item be the new current item, and the `SEND-TO-MOTOR` action instructs the ocular motor processor to move the eye to this object. When the text label for this object becomes available, the rule might fire again. However, if the text label for the object matches the sought-for label, the rule `TARGET-IS-LOCATED-BEGIN-MOVING-MOUSE` will fire instead. This rule disables both itself and the first rule from firing again by removing the `(STEP VISUAL-SEARCH)` item from the database, enables the next step in the procedure by adding a `STEP` item, and sends the manual motor processor an instruction to perform a right-hand movement using the `PLY` style on the `MOUSE` device, which generates zero-order control movements to position the cursor onto the target.

Applying this model to Nilsen's task requires estimating a parameter for how long it takes to recognize the text label for the digits, and where on the retina this recognition could be done. Hornof assumed that the digit recognition could be done only in the fovea, and required 200 ms per digit, a value that had been used for similar stimuli in models for other domains. EPIC was run in a simulated version of Nilsen's experiment, and predicted menu selection times were obtained. As shown in Figure 3, the serial search model seriously misfits the data by predicting a slope of about 380 ms/item, far larger than the observed value of about 107 ms. The reason for the discrepancy is that according to the EPIC architecture, the serial search model will require an estimated time of at least 200 ms for the perceptual process to identify the menu item, about 50 ms for a cognitive-processor production rule to fire to initiate the eye movement to the next item, and due to the savings from repeated similar movements, only somewhat more than about 50 ms for each eye movement. Even if the perceptual processing takes much less time, say only 100 ms, the oculomotor and cognitive times are still too long to produce a slope as shallow as 100 ms. In summary, data such as Nilsen's are extremely hard to reconcile with a model that assumes a strategy of sequentially fixating and deciding about each menu item separately.

Overlapping search model. Hornof next developed a more sophisticated model that more fully exploits the parallelism possible in EPIC. The relevant rules are shown in Figure 4. The production rule `SACCADE-ONE-ITEM` moves the eye from item to item as rapidly as possible. Before the eye moves on, the digit in the fovea gets started into the perceptual recognition "pipeline" mentioned above, and eventually the recognized property of the object gets deposited in visual working memory. Meanwhile the rule `STOP-SCANNING` functions as an independent "demon" waiting for an item to appear in visual working memory that matches the target. As soon as it does, this rule shuts down the scanning process, and enables the next rule, `MOVE-GAZE-AND-CURSOR-TO-TARGET,` which launches an eye movement and mouse cursor movement back to the matching object. Thus the search for the matching item is

conducted partially in parallel, since the process of recognizing the labels for different objects can be overlapped in time, both with each other, and with the eye movements required to move the fovea between the objects. Using the same parameters for the digit recognition availability and time, this overlapping search model predicts the values shown in Figure 3; the model predicts a slope of about 103 ms, which is an excellent match for the empirical value.

This model asserts that the eye can be scanned down the menu at high speed, with very little delay between saccades; empirical work on eye movements may disconfirm the latter assumption. If so, then other possible models might account for Nilsen's data instead. For example, a variation on the overlapping search model assumes that the menu items can be recognized somewhat outside the fovea, enabling a strategy of moving the eye only to every third item or so; when the desired item is detected, the eye and mouse movement would be made to it. This alternative model requires fewer eye movements that could be made more slowly, but still produces the overall fast menu selection times. In addition to developing these alternative models, Hornof is conducting additional modeling work on other conditions in Nilsen's data, with the goal being a unified account of menu selection in terms of visual

```
(*Enter-number*Get-next-digit
IF
((GOAL Enter number)
 (STEP Get next digit)
 (WM Next speech is ?prev)
 (AUDITORY SPEECH PREVIOUS ?prev NEXT ?next TYPE DIGIT CONTENT
?digit)
 (VISUAL ??? SHAPE FIVE-KEY)
 (MOTOR MANUAL PROCESSOR FREE))
THEN
((SEND-TO-MOTOR MANUAL PERFORM Peck ?digit)
 (DELDB (WM Next speech is ?prev))
 (ADDDB (WM Next speech is ?next))))
```

Figure 5. Production rule that enters each digit in the telephone operator task model. The output from the auditory recognition process identifies the key to be struck by the manual motor processor.

search and pointing mechanisms. Notice that an evaluation tool for visual layout should directly result from such models: two designs for the visual layout of the interface can be specified, and the corresponding EPIC models for the task can be run to determine which interface demands more visual work in the form of eye movements or perceptual delays.

Conclusions. This preliminary work illustrates how even an elementary aspect of using an interface can involve important and subtle aspects of the human performance architecture. Also, it shows how the quantitative parameter values built into and generated by EPIC impose powerful constraints on what strategies can serve as accurate models of cognitive processing. For example, it is possible to rule out the serial search strategy because EPIC determines that the absolute minimum times for moving the eye and making the decision for each item are on the order of 100 ms, leaving no time for perceptual processing of the items. This minimum time in turn is determined by EPIC's feature representation of movements, in which the similarity of the repeated eye movements resulting from the specific visual layout of the task means that they will take little time to prepare. A final quantitative result provided by EPIC's mechanisms concerns the linearity of the selection time function: the non-linear component contributed by the mouse movement time is not sufficiently large, given the specific layout of the menu, to produce a detectably non-linear selection-time function.

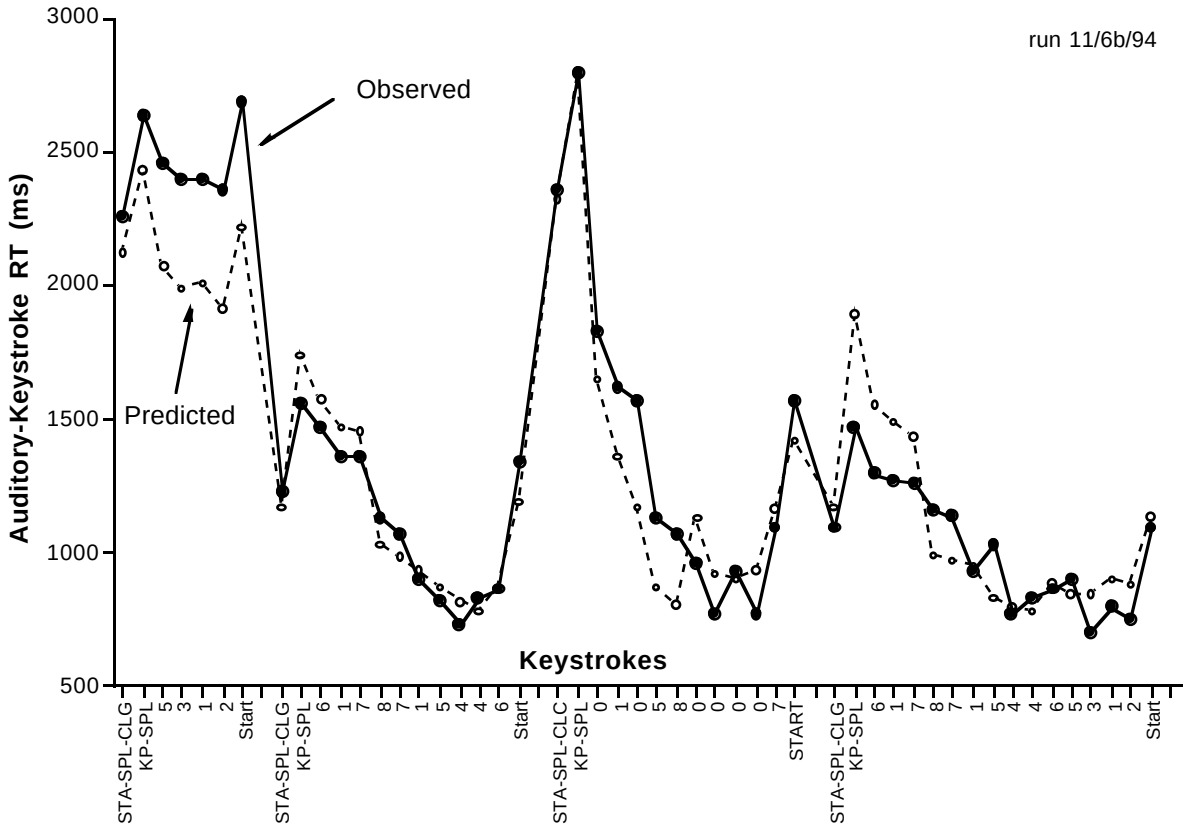


Figure 6. Observed and Predicted reaction times for keystrokes made in response to speech input. The keystrokes for four different task instances are shown in temporal order on the horizontal axis.

Auditorily-Driven Keyboard Data Entry

Another illustration that even elementary HCI tasks can involve considerable parallelism appears in modeling telephone operator tasks. Skilled telephone operators can listen to a string of digits spoken by a customer and enter them on a keypad while they are still being spoken. During such processing, the auditory perceptual processor, the cognitive processor, and the manual motor processor are operating simultaneously; the cognitive processor plays the role of mediator between the auditory and motor processors, feeding instructions to the motor processor as rapidly as the recognized digits become available from the perceptual processor, and as soon as the motor processor is ready to accept instructions for another keystroke movement preparation. As part of the work described in Kieras, Wood, and Meyer (1995a, b), this performance has been examined and modeled in detail.

Overlapping auditory and manual processing. The auditory processor produces a series of auditory working memory items that contain the recognized digits recoded as identities of the keypad keys, chained together to preserve the order in which they were heard. Figure 5 shows a production rule, `*Enter-number*Get-next-digit`, from a model used in Kieras, Wood, and Meyer (1995a, b). The rule uses these recoded digits to make the corresponding keystrokes in a "pipeline" fashion similar in spirit to John's (1988) model of transcription typing. Each recognized spoken digit is represented as a sequentially tagged item in auditory working memory, of the form `(AUDITORY SPEECH PREVIOUS ?prev NEXT ?next TYPE DIGIT CONTENT ?digit)`, where the variable `?digit`

represents a recoding supplied by the auditory perceptual processor that designates the physical target of the corresponding key.

As each digit arrives in working memory, the rule fires when the manual motor processor has begun executing the previous keystroke, then sends the keystroke command corresponding to the digit to the manual motor processor, and also updates a "pointer" in WM to the next speech item in auditory working memory to be processed. The rule also requires that before the digit can be typed, the shape of the center key on the keypad, the `FIVE-KEY`, must be in visual working memory to ensure that the target key is in view. The manual motor processor uses the code for the digit key to prepare the features for a peck-style movement to strike the key, and then initiates the prepared movement as soon as the hand is physically free to do so. The preparation process takes 0-100 ms depending on the similarity of the present movement to the previous movement, while the movement itself takes 50 ms to initiate followed by a minimum of 100 ms to physically execute. Thus, there is enough time during each keystroke execution to prepare the features for the next keystroke, assuming that the cognitive processor has provided the next keystroke instruction soon enough. If so, then a sequence of keystrokes can be made quite rapidly, on the order of 150 - 200 ms apart. However, if the auditory input is not supplied rapidly enough, the keystrokes will be slower, but exactly how much slower depends on the subtle details of the event timing.

Buffering effects. In the telephone operator task, the subtask of entering spoken digits is part of a larger task in which the operator must determine from the customer's speech what keys to press to indicate the billing category of the call and then entering the billing numbers. The operator first indicates the billing category by striking the `STA-SPL-CLG` key, and then strikes the `KP-SPL` key to signal that the billing number is about to be entered on the numeric keypad, and then enters the billing number digits. Performance in the whole telephone operator task has been modeled by Kieras, Wood, and Meyer (1995a, b), but a specialized EPIC model was used to explore some of the details of the auditory digit entry subtask. In this model, the operator waits until the customer speaks the first digit before striking the `STA-SPL-CLG` key, followed by the `KP-SPL` key, and then the digit keys are struck by the rule in Figure 5. Figure 6 shows the observed and predicted reaction times for each keystroke, measured from the auditory stimulus event defined as the stimulus for that keystroke. The observed reaction times are from a set of four task instances performed by a single operator; thus these are unaggregated, individual subject observations. This operator was observed on the videotape to strike all the keys with a single finger, making the peck movement style a clear choice for the model.

The values predicted from the EPIC model capture the general trend that the first several keystrokes are substantially delayed by the need for the previous keystrokes to be made before the first digit keystroke. Gradually, the digit keystrokes catch up because the customer is speaking the digits at a lower average rate than they can be typed. The shortest reaction times are where the processing has caught up to the point that there is no idle waiting time. The auditory recognition parameter can thus be estimated as the difference between these observed shortest reaction times and the predicted time for the cognitive and motor processing to produce these keystrokes. This parameter estimate, 400 ms, together with the above-described strategy, suffices to produce the observed complex profile of reaction times in response to the speech input. Given the single-observation quality of the data, the fit between predicted and observed reaction times is quite good.

Conclusions. The model shows that many of the keystrokes in the task are delayed more than the architecture requires, suggesting that performance could be speeded up by changing the workstation design in two ways: First, if the first two keystrokes could be eliminated or placed elsewhere in the task, performance would be speeded up and fewer digits would have to be buffered in working memory. Second, if the customer spoke the digits at a higher rate, or perhaps speech compression was used to produce the same effect, the task could in fact be done faster on the average. Given that in this task domain, saving a second of average task time is attributed with a considerable financial saving (Gray, John, & Atwood, 1993), the ability of EPIC to reveal these detailed aspects of performance is an important result.

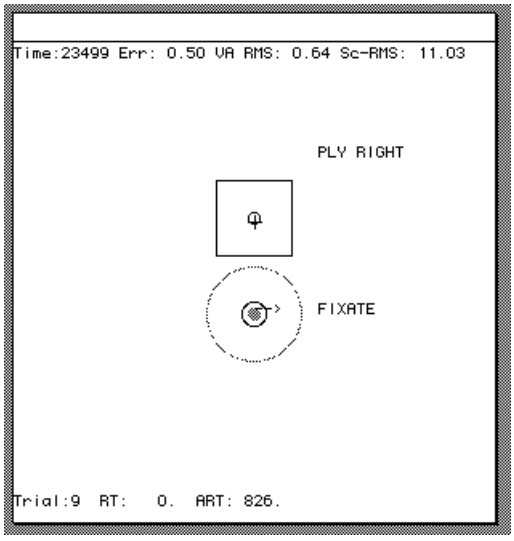


Figure 7. EPIC model display for the Martin-Emerson and Wickens task. The tracking task is in the upper square; the eye is fixated on the choice stimulus arrow appearing (not to scale) in the lower circle.

Towards a Model for Error Prediction

User errors have long been recognized as a serious problem in all technological systems, including computer user interfaces where errors are quite frequent (Shneiderman, 1992). Not only are user errors potentially dangerous, but errors typically require considerable time for the user to recover. Although this topic is critical both scientifically and practically, our theoretical understanding of error is seriously underdeveloped, especially in procedural tasks, which are the primary locus of errors while interacting with a computer system. The major contribution of research on human error to date has been several taxonomies for classifying the different types of error, but none of these taxonomies has been integrated with an adequate and comprehensive architecture for human cognition and performance. Since the available design principles for how to prevent human error consist of only a few crude guidelines (e.g. Norman, 1983), a valuable contribution to system design practice would be some analytic approach to determining whether a user interface design is especially error-prone, and where exactly errors are likely to appear. Perhaps a computational model of error can be developed by starting with the current EPIC architecture, introducing some unreliability into the architecture mechanisms, and then determining whether the observed misprocessing produces task errors of the type and frequency actually observed.

Working memory unreliability as a cause of errors. Working in our lab, Scott Wood has begun exploring some hypotheses about how EPIC might account for errors, using a task based on an experiment by Sellen (1986). In this task, subjects learned to produce responses to visually-presented digits; each response consisted of a long sequence of alphanumeric keystrokes. Sellen manipulated the similarity of the response sequences and how frequently each response was used; she observed many different types of errors, especially *capture errors,* in which the subject responds to a stimulus with a sequence that is similar to and occurs more frequently than the correct response. The basic hypothesis Wood has been exploring with EPIC is that errors in procedure execution may be due to unreliability in working memory. However, there are many possible ways for working memory to be unreliable, so the problem is to determine what form of unreliability could account for the observed errors and their relative frequency.

For example, Wood has determined that if working memory items simply disappear, then the resulting errors are almost always of a "halting" sort, in which performance simply stops, and has to be restarted or aborted. Note that even in this simple case, it is necessary for the model to have a set of rules for detecting and recovering from a procedure execution error. A more interesting hypothesis about the nature of memory unreliability is based on the observation that people usually make errors that in some sense are still appropriate to the task domain - for example, a pilot may press the wrong button, but will not start singing an aria, even though it is physically possible to do so. Thus, perhaps working memory unreliability is a result of memory items being confused with similar memory items. Wood is testing models of memory failure in which if a current memory item is very similar to an item that was recently in memory, then the current item might be corrupted into the previous item. The probability of memory corruption is higher the more items are present in working memory, and the most recent similar item is more likely to determine the result of the corruption.

Wood's preliminary results using this form of working memory unreliability are quite intriguing. Many corruptions do not yield observable errors, because the corrupted items can no longer affect the course of task execution. Quite often the corruptions cause the task procedure to halt or become stuck. The current model simply aborts the trial, but a more complex model would attempt to recover and continue, perhaps producing other forms of errors as well as correct (but delayed) responses. Many of the other errors are repetitions of part of the response, and some are omissions of one or more keystrokes. While the repetition errors are probably too frequent compared to empirical results, the capture errors agree with Sellen's results in being fairly common and being affected by the response frequencies: the high frequency responses are produced erroneously more often than the low frequency responses. Thus, a single source of unreliability, memory corruptions based on memory load, similarity, and recency, may produce a large variety of error types in procedural tasks. Further work will involve detailed comparison of empirical data on errors with different hypothetical mechanisms for error production in combination

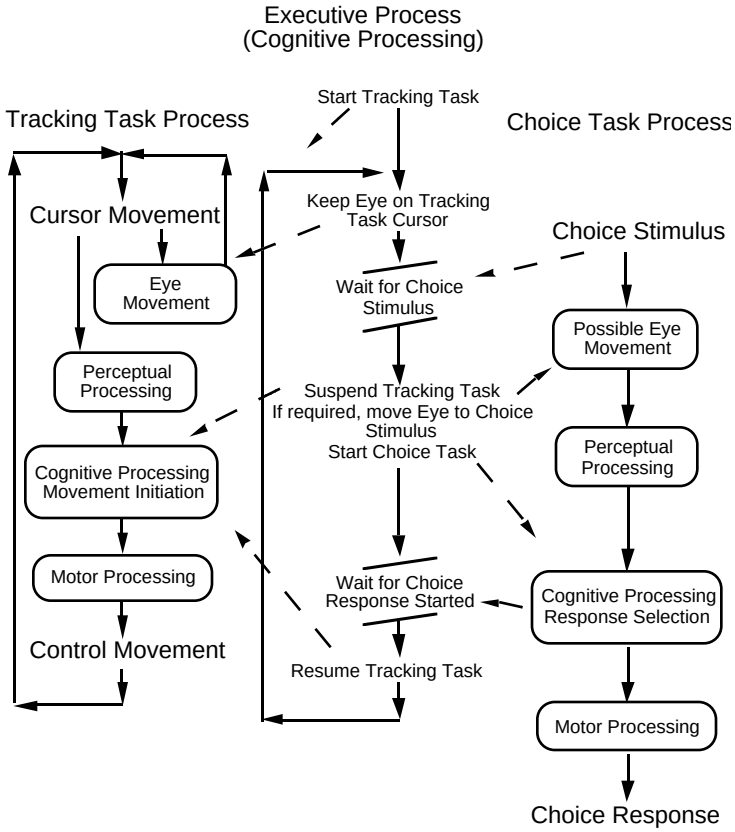


Figure 8. Flowchart of Lockout Model strategy for the Martin-Emerson and Wickens task. The tracking task is suspended whenever the choice task is executing.

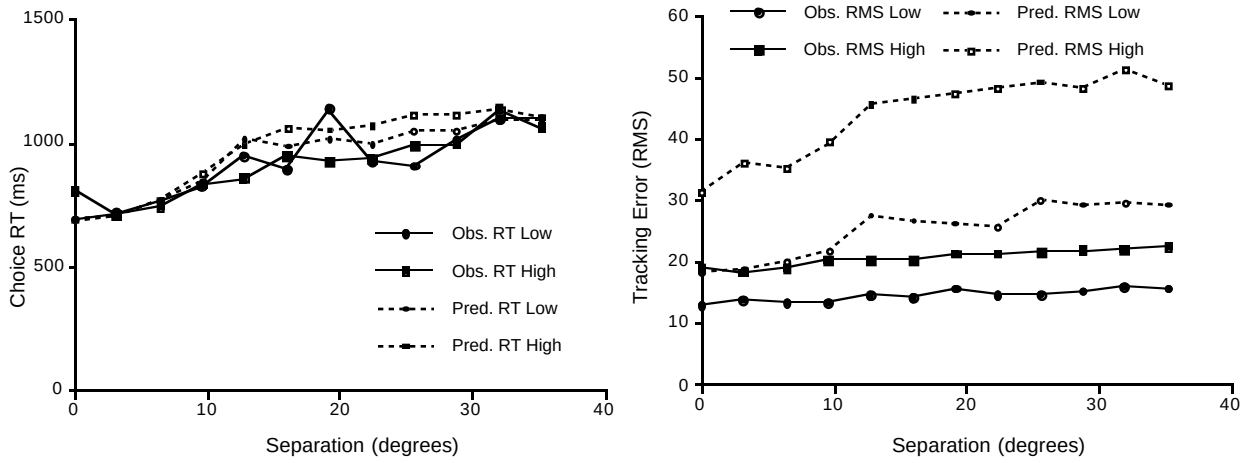


Figure 9. Predicted and observed effects of stimulus separation and tracking difficulty for the Lockout Model. Observed are large solid points and lines; predicted are small open points and dotted lines. The fit of the choice RT is satisfactory, but predicted tracking performance is far too poor.

with production-rule strategies for error detection and recovery. In this way, EPIC, an architecture that represents many aspects of correct performance, will also enable substantial progress towards a theory of human error in procedural tasks.

Conclusions. If this work is successful, it should be possible to take an EPIC model for performing a task correctly with a proposed interface design, and then simulate unreliable performance, and so determine at what points in task execution the user will be especially likely to make errors. By examining the model behavior, the interface designer could determine the causes of the errors, and then revise the interface to reduce their likelihood. Also the designer could propose interface displays and procedures that should help the user detect and correct the errors, and then add the corresponding strategy to the EPIC model. A new simulation run could then verify whether the interface revision or the proposed detection and recovery strategies indeed reduce the frequency or time cost of errors.

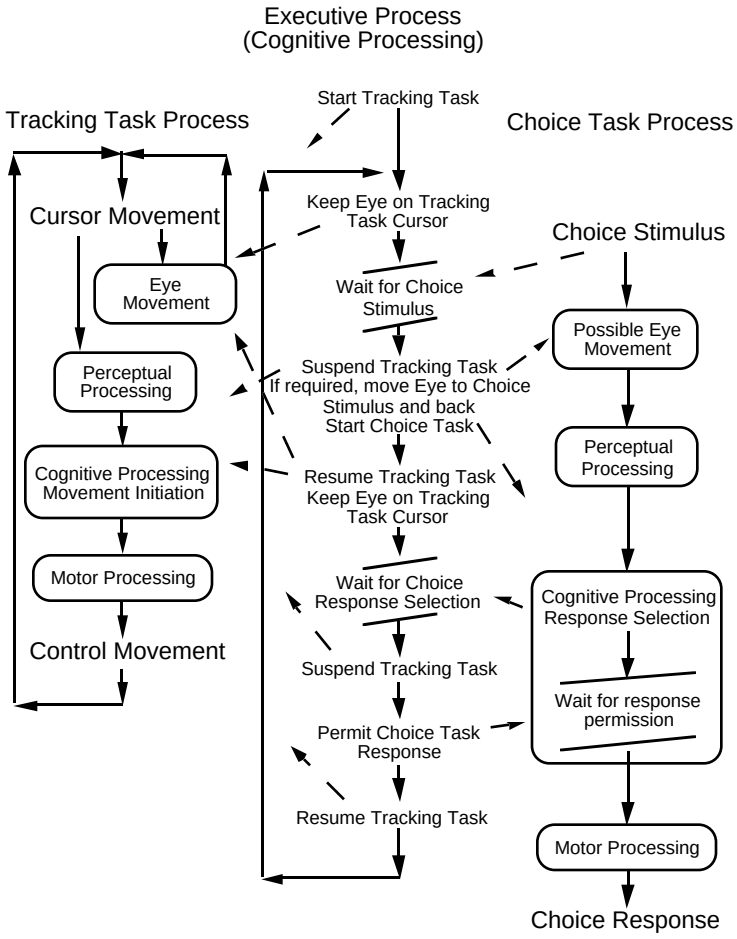


Figure 10. Flowchart of the Interleaved Model for the Martin-Emerson and Wickens task. The tracking task is executed while choice stimulus recognition and response selection go on, and is interrupted for the minimum necessary time.

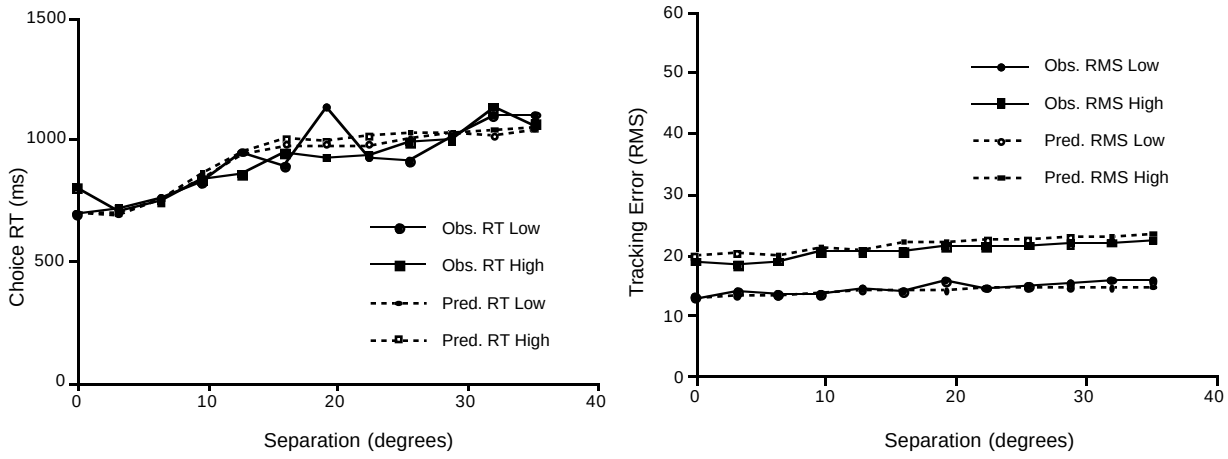


Figure 11. Predicted and observed effects of stimulus separation and tracking difficulty for the Interleaved Model. Observed are large solid points and lines; predicted are small open points and dotted lines. The fit is good for both choice RT and tracking error.

Eye Movements and Executive Control in a Simple Dual Task

Computers are used not just in desktop or office settings, but also in real-time contexts such as airplane cockpits, where many of the displays and controls are actually computer interfaces. As mentioned above, EPIC has been developed to deal with multiple-task situations, such as those in which a pilot has to track a target on one display and make decisions using the information in another display. The spatial distribution of visual information and the timing of eye movements play critical roles in determining performance in such situations. The remainder of the paper will describe two studies, one involving a simple form of this paradigm, and the other involving a complex form.

We have modeled some results obtained by Martin-Emerson and Wickens (1992) that are especially instructive about the role of eye movements and the use of visual information during dual tasks. Figure 7 shows the EPIC model display of the task. The display represents the visual environment of EPIC with the objects in their correct sizes and positions; the small gray circle marks the location and size of EPIC's fovea, currently on the choice stimulus, and the larger gray circle marks the boundary of the parafovea, a region of intermediate discriminative ability.

The compensatory tracking task uses the portion of the display in the upper box; a quasirandom perturbing force drives the cursor (the cross) away from the target (the small circle); the subject must manipulate a joystick with the right hand to keep the cursor centered on the target. Occasionally a stimulus appears in the choice stimulus area (the solid circle below the tracking box), which is either a left- or right-pointing arrow (not shown to scale in the display). The subject must respond by pressing one of two buttons with the left hand as soon as possible, while attempting to maintain the cursor on the target.

The major independent variable is the distance (in visual angle) between the tracking target and the choice stimulus, and a second independent variable is the difficulty of the tracking task. The two dependent variables are the reaction time for the choice task and a measure of tracking performance, namely the average RMS error in the tracking task, collected for a two-second period following the onset of the choice stimulus. The observed effects are that the choice reaction time increases with the angular distance between the target and the choice stimulus, but is

unaffected by tracking difficulty. The RMS error increases somewhat with the angular distance for both levels of tracking difficulty.

Our models for this dual-task situation assume that successful tracking requires the eye to be kept on the tracking cursor, and likewise, the eye must be moved to the choice stimulus in order to discriminate it. However, if the choice stimulus is close enough to the tracking cursor, parafoveal vision will be adequate to discriminate the stimulus without moving the eye. Hence the two tasks often, but not always, compete for use of the eye. Finally, because both tasks involve manual responses, they compete for access to the manual motor processor. We will illustrate how EPIC can be applied to this task with two models.

A simple lockout model of executive control. The Lockout Model uses a simple strategy, shown in flowchart form in Figure 8, that is consistent with traditional thinking about dual-task situations, namely the lower priority task is locked out (suspended) while the higher priority task is executed. The tracking task rules simply make a motor movement whenever the cursor is too far off the target. The executive process normally allocates control of the eye to the tracking task, where a production rule ensures that the eye makes a movement to the cursor anytime it is too far off. The oculomotor processor also can autonomously make small adjustments using perceptual information about object movements. When the choice stimulus appears, the executive process suspends the tracking task, activates the choice task, and then allocates control of the eye to the choice task, moving it to the stimulus if it is too far away to be discriminated. When the choice response has been initiated, the executive resumes the tracking task and returns control of the eye to the tracking task. In this way, using what we term *lockout scheduling*, the executive process allows only one task to be done at a time, ensuring that the choice task has priority over the tracking task, and that the eye and the manual motor processor are only used for one task at a time.

Unfortunately, this simple strategy does not fit all aspects of the data. Figure 9 shows the observed values for the choice reaction time and the tracking error together with the values predicted by the Lockout Model. Using a best estimate from the data of the perceptual recognition time for the choice stimulus, we can fit the choice reaction time data fairly well. There is no effect of tracking task difficulty since the choice task is given priority over tracking. The first few points are fairly flat, due to the parafoveal recognition of the choice stimulus. The upward slope of the curves at larger separations reflects the time required to move the eye. The fit of the simulated tracking data is extremely poor, however. The magnitude of the tracking error is seriously overpredicted, as are the effects of tracking difficulty and visual separation.

The Lockout Model cannot be made to fit the tracking data better by adjusting the relevant parameter values; it is already using the minimum plausible time estimates for all of the perceptual and motor parameters involved. Because the RMS error is measured for a brief (2 s) period of time starting with the onset of the choice stimulus, if tracking is suspended for too long during this period, the effect will be substantial. The Lockout Model suspends the tracking task for such a long time that considerable tracking error accumulates; it is simply too inefficient.

An interleaved model of executive control. In order to provide more efficient strategy, we constructed a second model, the Interleaved Model, in which the executive process overlaps the two tasks as much as possible; this strategy is shown in Figure 10. When the choice stimulus appears, the executive moves the eye to it, and then immediately begins to move the eye back, relying on the "pipeline" property of the visual system to acquire the stimulus and continue to process it, even after the eye has returned to the tracking cursor. The tracking task is suspended only while the eye is away looking at the stimulus arrow for the choice task. The model uses the same approach as in our PRP models for allocating control of the manual motor processor. When the choice task has chosen the response, it signals the executive, which again suspends the tracking task, allows the choice task to command the manual motor processor, and then resumes the tracking task right away. Thus, the same task priorities are honored, but the tracking task is interrupted as little as possible. The predictions from this model are shown in Figure 11. The choice reaction times are again well fit, but now the tracking task predictions are

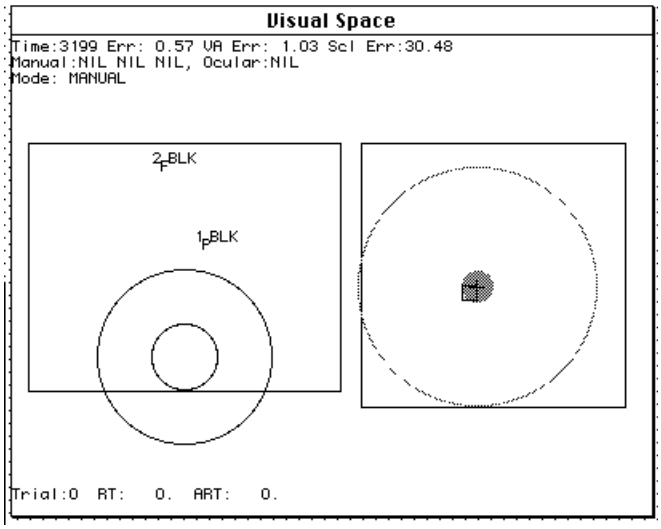


Figure 12. EPIC model display for the Ballas et al. task. The eye is on the tracking task on the right; two targets are moving down the tactical decision task display on the left.

extremely close as well. Since the executive allocates the eye and the manual motor processor to the tracking task for the maximum amount of time, the tracking task rules can squeeze in a few movements while the choice task is underway, resulting in substantially better tracking error than does the Lockout Model.

Conclusions. As mentioned in the introduction, the control of the eye has often been unappreciated, but it can clearly be critical in dual-task paradigms. A more subtle result is that subjects can and apparently do use highly refined strategies that can be surprisingly efficient for coordinating dual tasks.

An important general conclusion well illustrated by this particular modeling work concerns a common misunderstanding about computational models. They do not in fact have so many "degrees of freedom" that they can be made to fit any data at any time. Working within the fixed EPIC architecture sets powerful constraints. Given the basic lockout strategy, there are no parameter values or specific strategy details that would allow us to fit the data as a whole. The only way EPIC could be applied to fit the data is by assuming a fundamentally different strategy. Thus, a general conclusion (also observed in our earlier work) is that the exercise of seeking quantitatively accurate accounts of data within a fixed architecture is extremely informative about both the accuracy of the architecture itself and also the structure and requirements of the task.

A Complex Dual Task with Computer Automation

Our modeling work on the Martin-Emerson and Wickens task has laid the foundations for our current work on a more complex dual tracking/choice task. This task was developed by Ballas, Heitmeyer, and Perez (1992a, 1992b) to resemble a class of tasks performed in combat aircraft in which analyzing the tactical situation is partially automated by an on-board computer. To help the explanation, Figure 12 shows our EPIC model display for this task. The right hand box contains a pursuit tracking task in which the cursor (cross) must be kept on the target (small box); EPIC's eye is shown currently on the cursor. In an experiment reported by Ballas et. al., average tracking error data were collected during various phases of the experiment. The left-hand box contains the choice task, a tactical decision task in which targets (or "tracks") must be classified as hostile or neutral based on their

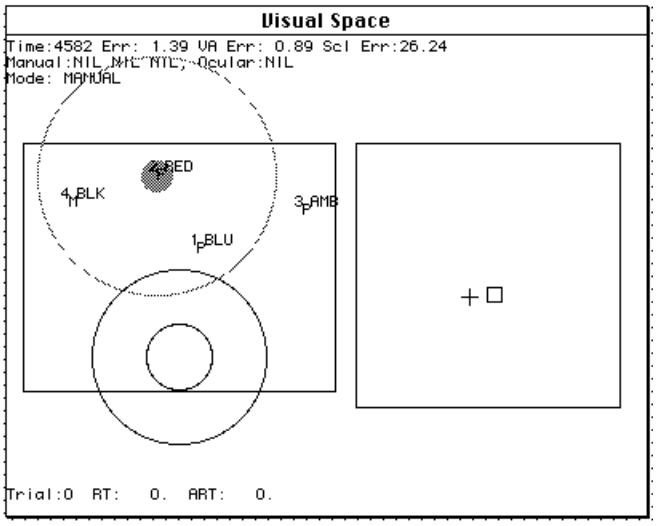


Figure 13. Automation deficit effect: After resuming the tactical task, the model inspects the objects out of order. The bottom-most target should have been processed first, but a different target was picked for first processing because order and priority information was not maintained while the tactical task was automated.

behavior. These targets represent fighter aircraft, cargo airplanes, and missile sites that move down the display as the subject's aircraft travels. Ballas et al. collected choice reaction time data during performance of the tactical decision task.

In the actual Ballas et al. (1992a) display, each type of target was coded by an icon; for simplicity, in the EPIC display they are represented instead by a code letter. A track number identifies each object. Targets appear near the top of the display, and then move down the display. After some time, the on-board computer attempts to designate

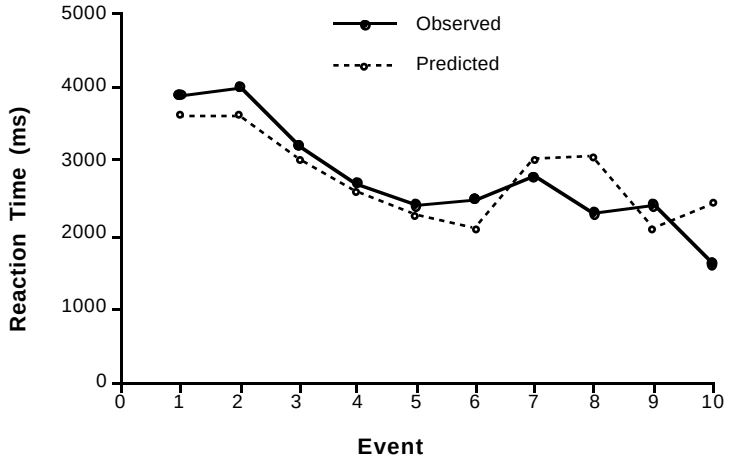


Figure 14. Automation deficit effect: Observed and predicted reaction times for events following resumption of the tactical task. Starting with Event 1, responses are delayed and then speed up as the tactical task catches up with the situation. Event 7 appears after the tactical task as been terminated and tracking has been resumed.

the targets, indicating the outcome by changing the target color from black to red, blue, or amber, which the EPIC display shows with a three-character abbreviation. If the target becomes red (hostile) or blue (neutral) the subject must simply confirm the computer's classification; the response is striking a key for the hostile/neutral designation followed by a key for the track number. If the target becomes amber, the subject must classify the target based on a set of rules concerning the target's behavior, and then type the hostility designation and track number. After the response, the target changes color to white, and then disappears from the display some time later. The basic dependent variable is the reaction time for the targets, measured from when the target changes color to when the first of the two response keystrokes are made.

Ballas et al. investigated different interfaces for the tactical task. The above description pertains to one of the four interfaces; the other three involved using a tabular display instead of the graphical radar-like display, and a touchscreen instead of a keypad. At this time we have modeled performance for only the one interface described above. Our future plan is to develop models that account quantitatively for the effects of the other display and response formats.

A performance deficit produced by automation. Ballas et al. examined the effects of *adaptive automation*. These effects arise when, from time to time, the tracking task becomes more difficult, and the on-board computer takes over the tactical task, signaling as it does so. The computer then generates the correct responses to each target at the appropriate time, with the color changes showing on the display as in the manual version of the task. Later, the tracking becomes easy again, the computer signals, and then returns the tactical task to the subject. Under such conditions, Ballas et al. observed an *automation deficit effect*, in which for a time after resuming the tactical task, subjects produced longer response times in the tactical task compared to their normal steady-state manual performance. This effect raises serious concerns about possible negative consequences of automation in combat situations; if the automation fails, the operator can lack situation awareness, and it might take a dangerously long time to "catch up."

A preliminary explanation for automation deficit. At present, our EPIC models for the Ballas task are still under development; this description of our results is subject to revision in later presentations of this work. From single-task performance, we have estimated the parameters for the basic perceptual encoding operations required in the tactical task, namely recoding the blue and red colors to the appropriate key, and recognizing the hostility of different kinds of targets, which takes considerably longer (more than a second). We have assumed that assessing the hostility of a target requires the target to be visually fixated, but that a target's color is available parafoveally, and color changes, like object onsets and offsets, are visible in peripheral vision.

Our current preliminary model is a simple one, structured much like the lockout model described above for the Martin-Emerson and Wickens task; we may need a more complex interleaved model to fully fit the data. When the tactical task is being done by the subject (as opposed to the on-board computer), the executive process allows the tracking task to run until it is time to work on the tactical decision task. In the meantime, the executive process simply notes the appearance of a target but continues tracking and waits for a detection of a color change in peripheral vision, or for a single target to get close to the ownship circle. The executive then suspends the tracking task, and allocates the eye to the tactical task. The tactical task follows a priority scheme in choosing which target to look at and process: targets with a designation (amber, red, or blue color) are first priority, followed by targets whose color has changed, followed by targets whose color is unknown, and finally followed by an undesignated (black) target that is close to the ownship circle. If no targets qualify, the tactical task terminates, and the executive resumes the tracking task. The eye is moved to the chosen target, and the appropriate response made when the perceptual information (color coding or hostility behavior) becomes available. Once the response is on its way to the manual motor processor, the process of choosing a new target begins in parallel with completion of the response. Thus, the tactical task has two major phases: choosing the stimulus to be processed, and selecting the response for the chosen stimulus.

Our current hypothesis about the source of the automation deficit effect is that when resuming the tactical task, the tactical task strategy must sort out a large number of targets, whereas during steady-state tactical task operation, the targets are handled as they appear. We assume that when the tactical task is automated, the subject does not bother to store any information about the state of the tactical display in working memory. Thus, when it is time to resume the tactical task, multiple targets are present on the tactical display; there is no record in working memory of which have changed colors or when the color changes occurred, and since most of targets are outside the parafovea, their color will not be available visually. If no colored targets are apparent, the tactical task strategy simply picks the first target to inspect at random. After moving the eye to it and waiting for the color to become available, the strategy processes the target as usual if it is red, blue, or amber. However, if the target is white or black, it cannot be processed, and so another target is picked at random according to a priority scheme. When all candidate targets have been dealt with, tracking is resumed, and future target changes are processed as they appear.

The automation deficit results because when the tactical task is being performed normally, targets are usually processed in the order that they change color, keeping the average reaction time to a minimum. In contrast, when the tactical task is resumed, multiple targets must be inspected, and no information has been kept on the order in which they have appeared or changed color (otherwise, the automation is of little value!). Thus the first few targets after task resumption are inspected in random order, so targets that changed first will have to wait longer on the average to be inspected than if they were processed as soon as they changed colors.

Figure 13 illustrates what can happen during tactical task resumption. Although Track 1 (a blue plane) was the earliest changing target, the strategy happened instead to pick Track 2 (a red fighter) to process first. After Track 2 is processed, the strategy will choose one of the remaining three targets to inspect next. If Track 4 is chosen, time would be wasted since a black target should not be processed yet; if Track 3 (an amber plane) is chosen, the other tracks would go unprocessed for a long time while the model waits for the hostility status of Track 3 to become apparent. Thus it may be a long time before Track 1 gets processed. As the model performs the task, it will catch up after some time, and targets will again be processed mostly in the order that they change. Thus, relative to steady-state performance, performance on the tactical task is poorer for some time following its resumption; temporarily, targets may take longer than normal to get processed.

Figure 14 shows some automation deficit results obtained with the current model in comparison to some data from Ballas et al. The graph shows the predicted and observed reaction times for each target, measured from the time that they become designated (change color), in the order that the targets become designated after the tactical task is resumed. Thus Event 1 corresponds to the first color-change of a target after task resumption, Event 2 to the second, and so forth. The overall quality of the fit is very good, but some discrepancies are systematic; additional modeling work to understand these discrepancies is underway

To understand how the current model accounts for the automation deficit effect in the Ballas et al. experiment, it is important to keep in mind that Ballas et al. systematically varied the temporal spacing of events in the tactical decision task; starting with Event 1, which is simultaneous with the signal to resume the tactical task, the events happen close together in time, and then thin out until by Event 6 usually only one target is present on the display; a similar pattern starts with Event 7. Thus the tactical processing is not evenly paced; there are periods of high workload followed by easy stretches.

The reaction time for Event 1 is long because the executive waits until the auditory resumption signal has been recognized and the current tracking actions have been completed before starting the tactical task; once started, the tactical task will frequently look at some other target first, further delaying the processing. Event 2 occurs only 1500 ms after Event 1, and so responding to it is also seriously delayed because it must wait for the delayed processing of Event 1 to be completed. On the average, Event 2 suffers less delay than it might because sometimes it gets processed before Event 1. Event 3 occurs 3000 ms after Event 2, Event 4 is 6000 ms after Event 3, and then Events 5, 6, and 7 occur after intervals of more than 10 seconds apart. This increasing event spacing allows the

tactical task to gradually catch up. By Event 6, the events are far enough apart that tracking is usually resumed. But when Event 7 occurs, tracking has been in progress, and the events are closely spaced again. Thus the reaction time to Event 7 is increased, and subsequent events suffer from the processing delays that gradually dissipate with the increasing event spacing. However, because the state of the display is being monitored at the time of Event 7, the tactical task is able to find its targets much more quickly than is the case at Event 1. Thus the effects of event spacing are more serious if the tactical display monitoring is just being resumed than if it was ongoing. Ballas et al. defined the automation deficit effect as the difference between the Event 1 reaction time and the reaction time for a matched subsequent event, namely Event 7. The current models predict this measure fairly accurately; fitting the remaining event times is the current goal.

Relationship to elementary dual-task phenomena. Modeling this task has revealed a remarkable continuity with our earlier modeling work with EPIC on the Psychological Refractory Period (PRP) task mentioned before (Meyer & Kieras, in press-a, b). The PRP task consists of two overlapping choice-reaction time tasks; the major effect is that responding to the second task is delayed while the first response is being made. As the time between the two stimuli is increased, the reaction time to the second task declines towards its single-task value. We chose the PRP effect as the first phenomenon to address with EPIC because it was the simplest laboratory version of a dual-task paradigm, and appeared to be a good starting point. However, even the complex simulated cockpit task of Ballas et al. produces PRP effects; that is, the declining reaction time pattern for the first six events in Figure 14 is a kind of PRP effect; the initial slow responses that gradually speed up as the stimulus spacing increases are due to exactly the same factors that govern the PRP effect in simpler laboratory paradigms. In other words, the automation deficit effect is a form of PRP effect. Our thorough understanding of PRP effects enabled by this earlier modeling work has allowed us to account for performance of this realistically complex task in quantitative detail.

Conclusions. If our current explanation for the automation deficit is borne out by our more complete and accurate models, there may be some important implications for display and task design. For example, according to this hypothesis, resuming the tactical task could be done more efficiently if it is possible to easily detect the highest-priority object on the display at that time. That is, suppose the first-changed object currently on the display was coded by making it blink, which would be salient in peripheral vision. Then the subject could simply look at the blinking object in order to ensure that the objects were processed in priority order. Alternatively, the automated version of the task could use a different, less salient, way of representing its activity, so that the subject could still profitably monitor for the same perceptual events that are important in the manual version. Not only does the EPIC architecture supply a theoretical framework in which such issues can be explored and resolved in rigorous detail, but also EPIC models can be used to evaluate and predict the effects the design changes implied by the explanations.

GENERAL CONCLUSIONS

EPIC is a computational architecture for constructing models of human cognition and performance that represent the contributions and interactions of perceptual and motor mechanisms as well as cognition in determining the time course of task execution. The examples presented in this paper illustrate how EPIC can be applied to a variety of situations in which humans interact with computers, both at the level of elementary interactions such as menu operation and data entry, and at the level of high-speed concurrent execution of multiple display-intensive tasks. By accounting for empirical data with high accuracy in an architectural framework, models constructed with EPIC provide explanations for task phenomena with a clarity and precision far beyond the informal theorizing normally deployed in the HCI field. Further work with EPIC should lead to a comprehensive and predictive theoretical account of human performance in complex high-performance multimodal tasks.

At this point, EPIC is a research system, and certainly is not ready for routine use by system or interface designers. However, there is a technology transfer precedent: earlier work with the CCT production rule models for HCI (Bovair et al., 1990) led to a practical interface design technique (Kieras, 1988; John & Kieras, 1994).

Likewise, as the EPIC architecture stabilizes and experience is gained in applying it to human-system analysis problems, we should be able to devise a simplified approach that will enable designers to apply EPIC to develop improved human-system interfaces.

Our experience with the EPIC architecture also suggests some meta-level conclusions about the role of cognitive modeling in the science and engineering fields of human performance and human-system interaction:

- Computational models that are based on human information-processing can usefully predict details of human performance in system design and evaluation situations.

- Developing and applying a cognitive model to task situations relevant to real design problems is a demanding test of cognitive theory; if the theory successfully represents important properties of human abilities, it should in fact be useful in practical settings.

- Powerful constraints are imposed by quantitatively fitting fixed-architecture models to detailed performance data. Thus, working with computational models is not necessarily an arbitrary exercise.

In short, a comprehensive, detailed, and quantitative theory of human cognition and performance is the best basis for applied cognitive psychology. Rather than relying only on general psychological principles, or brute-force application of experimental methodology, system design can be best informed by using a theory that addresses phenomena at the same level of detail as design decisions require. It really is true that "Nothing is more useful than a good theory!"

ACKNOWLEDGEMENT

This work was supported by the Office of Naval Research Cognitive Sciences Program under grant N00014-92-J-1173 to the authors. Thanks are due to James Ballas of NRL for his generous assistance in making his task software available and supplying new analyses of his data.

REFERENCES

Anderson, J. R. (1976). *Language, memory, and thought*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Anderson, J. R. (1993). *Rules of the mind.* Hillsdale, N.J.: Erlbaum.

Bovair, S., Kieras, D. E., & Polson, P. G. (1990). The acquisition and performance of text editing skill: A cognitive complexity analysis. *Human-Computer Interaction*, **5**, 1-48.

Ballas, J. A., Heitmeyer, C. L., & Perez, M. A. (1992a). Direct manipulation and intermittent automation in advanced cockpits. Technical Report NRL/FR/5534--92-9375. Naval Research Laboratory, Washington, D. C.

Ballas, J. A., Heitmeyer, C. L., & Perez, M. A. (1992b). Evaluating two aspects of direct manipulation in advanced cockpits. In Bauersfeld, P., Bennett, J., and Lynch, G., *CHI'92 Conference Proceedings: ACM Conference on Human Factors in Computing Systems*, Monterey, May 3-7, 1992.

Card, S. K., Moran, T. P., & Newell, A. (1983). *The psychology of human-computer interaction*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Gopher, D., & Donchin, E. (1986). Workload: An examination of the concept. In K.R. Boff et al. (Eds.), *Handbook of perception and performance*, Vol. II (pp. 41.1-41.49). New York: Wiley.

Gopher, D. (1993). Attentional control: Acquisition and execution of attentional strategies. In D. E. Meyer & S. Kornblum (Eds.), *Attention and performance XIV*. Cambridge, MA: M. I. T. Press.

Gray, W. D., John, B. E., & Atwood, M. E. (1993). Project Ernestine: A validation of GOMS for prediction and explanation of real-world task performance. *Human-Computer Interaction*, **8**, 3, 237-209.

Hallett, P. E. (1986). Eye movements. In K.R. Boff, L. Kaufman, & J. P. Thomas (Eds), *Handbook of perception and human performance*. Volume 1, 10-1 - 10-112. New York: Wiley.

John, B. E. (1988). *Contributions to engineering models of human-computer interaction.* Doctoral dissertation, Carnegie Mellon University.

John, B. E. & Kieras, D. E. (1994). *The GOMS family of analysis techniques: Tools for design and evaluation.* Carnegie Mellon University School of Computer Science Technical Report No. CMU-CS-94-181. Also appears as the Human-Computer Interaction Institute Technical Report No. CMU-HCII-94-106.

Kieras, D. E. (1988). Towards a practical GOMS model methodology for user interface design. In M. Helander (Ed.), *Handbook of Human–Computer Interaction* (pp. 135–158). Amsterdam: North–Holland Elsevier.

Kieras, D. & Meyer, D. (1995). Predicting human performance in dual-task tracking and decision making with computational models using the EPIC architecture. Proceedings of the *First International Symposium on Command and Control Research and Technology*, National Defense University, June. Washington, D.C.: National Defense University.

Kieras, D. E., Wood, S. D., & Meyer, D. E. (1995a). Predictive engineering models using the EPIC architecture for a high-performance task. In *Proceedings of CHI*, 1995, Denver, Co, USA, May 7-11, 1995. New York: ACM.

Kieras, D.E., Wood, S.D., & Meyer, D.E. (1995b). Predictive Engineering Models Based on the EPIC Architecture for a Multimodal High-Performance Human-Computer Interaction Task. (EPIC Tech. Rep. No. 4, TR-95/ONR-EPIC-4). University of Michigan, Electrical Engineering and Computer Science Department, Ann Arbor, Michigan.

Kristofferson, A. B. (1967). Attention and psychophysical time. In A. F. Sanders (Ed.), *Attention and performance* (pp. 93-100). Amsterdam: North-Holland Publishing Co.

Laird, J., Rosenbloom, P, & Newell, A. (1986) *Universal subgoaling and chunking*. Kluwer Academic Publishers: Boston.

Martin-Emerson, R., & Wickens, C. D. (1992). The vertical visual field and implications for the head-up display. *Proceedings of the 36th Annual Symposium of the Human Factors Society.* Santa Monica, CA: Human Factors Society.

McMillan, G. R., Beevis, D., Salas, E., Strub, M. H., Sutton, R., & Van Breda, L. *Applications of human performance models to system design.* New York: Plenum Press, 1989.

Meyer, D. E., & Kieras, D. E. (in press-a). Computational modeling of human multiple-task performance: I. The EPIC architecture and strategic response-deferment model. *Psychological Review*.

Meyer, D. E., & Kieras, D. E. (in press-b). Computational modeling of human multiple-task performance: II. Executive scheduling strategies and psychological refractory period phenomena. *Psychological Review*.

Meyer, D. E., Kieras, D. E., Lauber, E., Schumacher, E., Glass, J., Zurbriggen, E., Gmeindl, L., & Apfelblat, D. (in press). Adaptive executive control: Flexible multiple-task performance without pervasive immutable response-selection bottlenecks. *Acta Psychologica*.

Norman, D.A. (1983). Design rules based on analyses of human error. *Communications of the ACM*, 26(4), 254-258.

Norman, D. A., & Shallice, T. (1986). Attention to action: Willed and automatic control of behavior. In R. J. Davidson, G. E. Schwartz & D. Shapiro (Eds.), *Consciousness and self-regulation, Vol. 4.* New York: Plenum Press.

Nilsen, E. L. (1991). Perceptual-Motor Control in Human-Computer Interaction. (Tech. Rep. No. 37). Cognitive Science and Machine Intelligence Laboratory, University of Michigan, Ann Arbor, Michigan.

Rosenbaum, D. A. (1980). Human movement initiation: Specification of arm, direction, and extent. *Journal of Experimental Psychology: General,* **109**, 475-495.

Rosenbaum, D. A. (1991). *Human motor control*. New York, Academic Press.

Sears, A., and Shneiderman, B. (1994). Split Menus: Effectively Using Selection Frequency to Organize Menus. *ACM Transactions on Computer-Human Interaction,* **1**(1), 27-51.

Sellen, A. J. (1986). An experimental and and theoretical investigation of human error in a typing task. Unpublished Master's thesis, University of Toronto.

Shneiderman, B. (1992). *Designing the User Interface: Strategies for effective human-computer interaction*. Reading, Massachusetts: Addison-Wesley.

Welford, A. T. (1952). The "psychological refractory period" and the timing of high speed performance - A review and a theory. *British Journal of Psychology,* **43**, 2-19.

Wickens, C. D. (1984). *Engineering psychology and human performance*. Columbus, OH: Charles F. Merrill.